

CLIOQUERY: Interactive Query-oriented Text Analytics for Comprehensive Investigation of Historical News Archives

ABRAM HANDLER, Department of Information Science, University of Colorado, Boulder, USA
NARGES MAHYAR and BRENDAN O'CONNOR, Manning College of Information and Computer Sciences, University of Massachusetts, Amherst, USA

Historians and archivists often find and analyze the occurrences of query words in newspaper archives to help answer fundamental questions about society. But much work in text analytics focuses on helping people investigate other textual units, such as events, clusters, ranked documents, entity relationships, or thematic hierarchies. Informed by a study into the needs of historians and archivists, we thus propose CLIOQUERY, a text analytics system uniquely organized around the analysis of query words in context. CLIOQUERY applies text simplification techniques from natural language processing to help historians quickly and comprehensively gather and analyze all occurrences of a query word across an archive. It also pairs these new NLP methods with more traditional features like linked views and in-text highlighting to help engender trust in summarization techniques. We evaluate CLIOQUERY with two separate user studies, in which historians explain how CLIOQUERY's novel text simplification features can help facilitate historical research. We also evaluate with a separate quantitative comparison study, which shows that CLIOQUERY helps crowdworkers find and remember historical information. Such results suggest possible new directions for text analytics in other query-oriented settings.

CCS Concepts: • **Human-centered computing** → **Interactive systems and tools**;

Additional Key Words and Phrases: Digital humanities, history, interactive text analytics, user interfaces

ACM Reference format:

Abram Handler, Narges Mahyar, and Brendan O'Connor. 2022. CLIOQUERY: Interactive Query-oriented Text Analytics for Comprehensive Investigation of Historical News Archives. *ACM Trans. Interact. Intell. Syst.* 12, 3, Article 22 (July 2022), 49 pages.
<https://doi.org/10.1145/3524025>

1 INTRODUCTION

Newspaper archives are fundamental resources for historians, librarians, and social scientists [2, 20], because they offer a detailed primary source record of how social processes evolve across time [107]. For instance, social researchers have used news archives to examine vital questions such as why the United States abolished slavery [50] and how different jurisdictions slowed the

The reviewing of this article was managed by associate editor Liu, Shixia.

Authors' addresses: A. Handler, Department of Information Science, University of Colorado, Boulder, 1045 18th Street, Boulder, CO, 80309; email: abram.handler@colorado.edu; N. Mahyar and B. O'Connor, Computer Science Building, 140 Governors Drive, Amherst, MA 01003; emails: nmahyar@cs.umass.edu, brenocon@cs.umass.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.

2160-6455/2022/07-ART22 \$15.00

<https://doi.org/10.1145/3524025>

spread of the 1918 flu [87]. While historians are known to use archives in different ways (e.g., sequential browsing [2]), prior work reports that historians often look for “specific keywords” [2, p. 2] in newspaper corpora. For instance, scholars in history and social science journals describe tracking down and reviewing occurrences of words such as “William Benbow” [110], “Frances Maule” [125], “watermelon” [9], “Japanese beetles” [121], “refugee” [106], “Loving” [67], and “race suicide” [83] in news archives to help answer questions about society. In this work, we describe such search terms (e.g., “William Benbow”) as **queries** and we describe each exact occurrence of a query in an archive as a query **mention**. (Section 10 discusses possible improvements to exact string matching.) We then describe the task of locating query mentions as **mention gathering**, and the closely related task of reviewing and drawing conclusions from query mentions within the context of surrounding text as **mention analysis**.¹ We formally define these terms and tasks in Section 3.1.

Much prior work in interactive text analytics (Section 2) does not focus on helping people investigate mentions of a query word in context. Instead, prior systems (designed for different use cases) focus on the analysis of other latent and observable textual units, such as topics [78], events [82], document metadata [42], clusters [12], interrelated entities [53], or thematic hierarchies [13]. Using terminology from Chuang et al. [24], who articulate best practices for text analytics, because such systems do not use query words in context as their central “unit of analysis,” they offer “visual encodings,” “modeling decisions,” and interactions that are poorly “aligned” to the “tasks, expectations, and background knowledge” of historians and archivists.

This misalignment means that prior text analytics systems have concrete downsides for mention gathering and analysis. For example, some prior systems focused on helping people analyze high-level text units such as temporal trends in word use (e.g., ThemeRiver [61]) do not show query words in underlying text. Similarly, other systems offer only indirect and incomplete access to query words in context, via extraneous mediating abstractions. We detail these limitations in Section 2.3.

However, in practice, social researchers such as Shinozuka [121] and others [9, 67, 83, 106, 110, 125] do not report using specialized text analytics systems. Instead, these experts describe using traditional keyword document search engines like ProQuest [109] to analyze query words in corpora. (We use *corpus* and *archive* interchangeably; we assume the corpus is an archive.) Traditional keyword document search tools return relevance-ranked document lists in response to a free-text query. Because they are widely used in historical practice [2, 20, 110, 125], we propose they are baselines for mention gathering and analysis (Section 2.2.1).

Yet keyword document search tools also have limitations for finding and analyzing query words in context. First, because almost all words are very rare (a well-known property of text [142]), any given query will very likely appear only a small number of times within a document. This means that people reviewing a ranked document list will have to examine many passages within documents that do not directly mention their query term. While search within document features (e.g., `control + F` in Chrome [27]) can certainly help, gathering and analyzing query words in context using a keyword document search system still requires opening each article in its own window or tab,² locating mentions within the article, reading passages that mention the query, and integrating information from such passages with existing knowledge, before moving on to

¹Using terminology from prior work [108], it is possible to interpret mention gathering as a kind of information foraging and mention analysis as a kind of sensemaking.

²Showing a single document in a single window or tab is a common interface pattern. It is employed, for instance, in the Overview [13] and Jigsaw [126] document viewers, and in traditional search user interfaces, which often link to individual documents from a main search engine results page [31, Section 6.3].

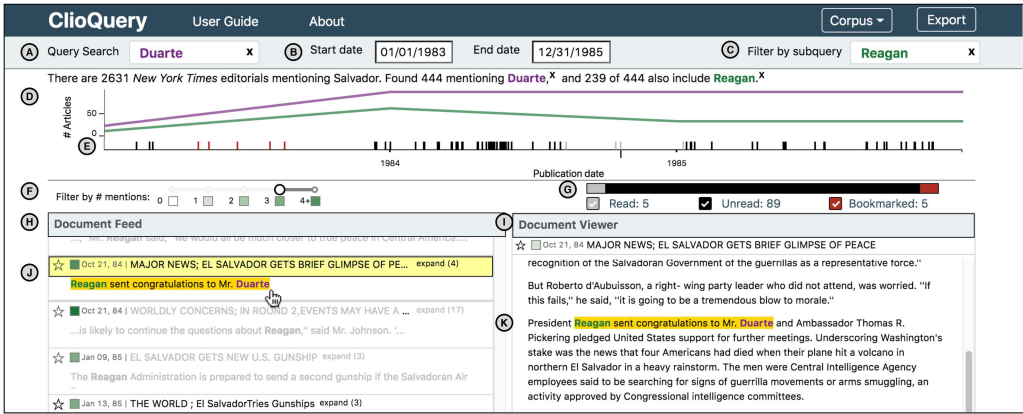


Fig. 1. CLIOQUERY, an interactive text analytics system for helping historians investigate queries in news archives. Features (letters A to K) include: (D) a Time Series View showing the frequency of a user's query through time, (H) a linked Document Feed showing a skimmable query-oriented summary of every mention of the query in the corpus, and (I) a linked Document Viewer showing a selected news story, with text from the query-oriented summary highlighted in yellow. Section 4 describes the full system, explaining each feature.

Table 1. This Work Presents Four Separate User Studies with Historians and Archivists

User study	Num. participants	Total hours
Needfinding study to guide system design (Section 3)	5	4.5
Expert interview study to evaluate CLIOQUERY features (Section 5)	5	5
Field study to test CLIOQUERY in the wild (Section 7)	2	5
Quantitative comparison study (Section 8)	121	40.3

Section 3 describes institutional approval. Tables in the Appendix describe the backgrounds of participants in greater detail.

the next document in the corpus. This means that people must context switch across stories as they perform a multi-step process to gather and analyze query mentions in context, keeping track of information from one document as they jump to the next (see Figure 2). Navigating between documents is thought to impose cognitive costs in keyword document search tools [46, 141], and context switching across views is thought to impose cognitive costs in visual analytics.

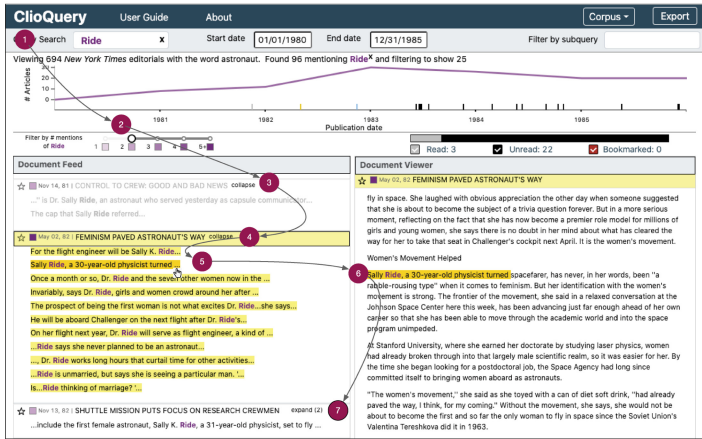
Noting the importance of historical investigation and the limitations of existing tools (for mention gathering and analysis), we propose the CLIOQUERY³ text analytics system to help historians in their work investigating query words in an archive. Unlike prior tools, which focus on the analysis of other textual units such as topics or hierarchies (Section 2.3), CLIOQUERY is designed and built to help people analyze query words in context, reflecting the needs and practices of historians and archivists. Creating tools around the “tasks, expectations and background knowledge” of end-users is believed to be a best practice in text analytics [24].

We both worked with and studied historians and archivists to prototype CLIOQUERY Table 1. This process revealed intertwined technical and design requirements for our system (Section 3). First, working with historians revealed the importance of comprehensive review in historical research (Section 3.3.2). Thus, CLIOQUERY includes a novel Document Feed feature, which uses **natural language processing (NLP)** techniques to show a comprehensive

³Clio is a common prefix (e.g., ClioVis [16]) implying a connection with history.



(a) A user investigates Sally Ride by performing mention gathering and analysis using the *New York Times* web archive [96], a baseline keyword document search interface (Section 2.2.1). They first (1) click the top headline on the search engine results page (left) in order to (2) open a document in a new tab (shown on the right) and then (3) scroll down to locate mentions of “Ride” in the linked news story. The user reads and analyzes these mentions and then (4) context switches to the second document by clicking the second headline on the results page. This (5) opens a new story in a new tab. For this second document, they (6) use a search in document feature [25] (i.e., Control+F) to help locate mentions of “Ride” within the story.



(b) A user (1) searches for “Ride” using CLIOQUERY and (2) sets the filter-by-count slider to limit results to stories with at least two mentions of “Ride.” The user then clicks the expand/collapse button on two news stories (3 and 4) to review all mentions of “Ride” from each story in the Document Feed. They then (5) click one shortened sentence mentioning “Ride” (6) to read it within the context of the full original document in the linked Document Viewer, with help from automatic in-text highlighting. The user then prepares to (7) click expand to review additional mentions of Ride in the next story.

Fig. 2. Reviewing mentions of U.S. astronaut Sally Ride in *The New York Times*, using CLIOQUERY (bottom) and a keyword document search tool (top). This particular example comes from our field study (Section 7), where one historian commented on the advantages of the CLIOQUERY interface over a baseline keyword document search system. “What can I do here [with CLIOQUERY] that I can’t do there [with New York Times search]?” she said. “It’s exploring this left-hand Document Feed.” Where CLIOQUERY facilitates quick and comprehensive review of all query mentions, the keyword document search tool requires the user to read unnecessary passages and context switch across documents.

query-focused summary of every single mention of a user's query term across a corpus. Similarly, because we found that historians require transparency and contextual information to interpret evidence, CLIOQUERY's novel Document Feed is presented alongside a more traditional linked full-text Document Viewer, which is designed to quickly and transparently show text from the summary within the context of full-length documents. Finally, because temporal analysis is crucial to historians, CLIOQUERY also includes an interactive Time Series View to provide an overview of a query through time. Together, through these and other features (Section 4), CLIOQUERY offers a text analytics system organized around the analysis of query mentions in context.

In total, our work offers the following:

- **A synthesis of extensive prior research in text analytics** (Section 2). In reviewing prior work on interactive analysis of text across time, we found that many efforts from the NLP, HCI, and Visualization communities focus on offering overviews of corpus contents. Such overviews might help users formulate queries, but are not designed for the query-oriented tasks of mention gathering and analysis (when the user already knows what to search for).
- **An investigation into user needs and requirements** (Section 3). To build our tool, we translated prior research on historians' information-seeking behavior into concrete guidelines for system design. We also validated and contextualized prior work by conducting five needfinding interviews with historians and archivists, while gathering feedback on early prototypes. This process revealed a need for transparency, trustworthiness, context, and comprehensiveness in archival tools, which might inform future work on historical search [123, 124] and text summarization [35, 97].
- **The CLIOQUERY system** (Section 4). CLIOQUERY is an open-source, text analytics system designed to help historians find and analyze query mentions in context. The system combines novel query-focused summarization methods with more traditional features such as linked views, in-text highlighting, and time series plots to help experts quickly, comprehensively, and transparently find and review query mentions across an archive. Code for the system is available on GitHub: <https://github.com/AbeHandler/ClioQuery>.
- **An evaluation of specific CLIOQUERY features** (Sections 5 and 6). To test the utility and usability of CLIOQUERY, we conducted an expert interview study with five social researchers, who used the system to answer a historical question from news archives. After methodically coding qualitative feedback, we learned that many experts found CLIOQUERY's skimmable, query-focused summaries useful, because they condensed documents to facilitate quick review of query mentions. We also learned that linking summary text with underlying source documents using in-text highlighting was essential, because it offered necessary context for interpreting summary output.
- **An evaluation of CLIOQUERY in the wild** (Section 7). To test CLIOQUERY in a realistic setting, we deployed the system in a field study with two historians who used CLIOQUERY to answer questions from their own research. In comparing experiences with CLIOQUERY to prior experiences with keyword document search systems, one historian explained how CLIOQUERY reduced their reading burden, and another explained how text summarization features facilitated rapid mention gathering and analysis.
- **A quantitative comparison with keyword document search tools** (Section 8). We conducted a quantitative crowd study to directly compare CLIOQUERY with baseline keyword document search systems. In the study, we observed that participants who used CLIOQUERY to complete a reading comprehension task modeled on a real historical research

question correctly answered significantly more reading comprehension questions than participants who completed the same task using a keyword document search system.

We conclude by discussing our findings (Section 9), reviewing limitations and future work (Section 10), and describing possible applications of features and ideas from CLIOQUERY in other query-oriented settings, beyond historical research (Section 11).

2 RELATED WORK

Historians sometimes gather and analyze mentions of specific query words in archives (Section 1). However, much prior work from the HCI, Visualization, and NLP communities focuses on helping people gain high-level overviews of large bodies of text. We review this overview-oriented literature in Section 2.1. In Section 2.2, we also review another literature on search-based systems, which focus on retrieving text from a corpus in response to a user query. This search-based approach seems better suited to mention gathering and analysis, as search-based systems can help historians find and review query mentions in a corpus. Much evidence (Section 1 and Table 2) also suggests that search-based systems are central to contemporary historical practice.

In presenting prior work, we emphasize common user interface design patterns [132] shared among multiple prior systems. Interface design patterns are “concrete bundles of components” [133] that help a user achieve some task. We say that *overview design patterns* (Section 2.1) help people survey the contents of archives, and that *search design patterns* (Section 2.2) help people query for specific selections from a body of text. Table 2 offers a summary of major design patterns from prior work. Some individual systems (e.g., Expedition [123]) may implement both overview and search patterns. Finally, we conclude this section by discussing CLIOQUERY within the context of prior work (Section 2.3).

2.1 Overview Design Patterns

2.1.1 Word Clustering. Because people often can not review every document in a large corpus, many prior text analytics tools such as Termite [23], TIARA [78], Overview [13], RoseRiver [33], TextFlow [32], Serendip [1], HierarchicalTopics [38], and ConVisIT [68] try to suggest overall themes in a body of text by identifying and displaying groups of thematically related words in a user interface. We describe this approach as the word clustering design pattern.

Many systems that implement the word clustering pattern are based on prior work from NLP, information retrieval, and text mining, focused on identifying and representing patterns of co-occurring words using methods such as topic models [10] and word embeddings [94].⁴ Researchers in HCI and Visualization extend this work by considering how to present such patterns in a graphical interface; some systems show changes in cluster patterns across time [32, 33, 78] (e.g., Figure 3(a)), others do not show time-based topics [13, 23]. Because automatic clusters may not match human mental models of a corpus, one line of work investigates human-in-the-loop techniques, which allow people to modify word clusters through interactions with a GUI [12, 68, 69, 73, 74, 103, 120].

Word clustering has a clear role in historical research. In query-oriented settings, clustering methods may help people formulate queries they had not considered [135]. Moreover, specialized and computationally oriented digital humanists [96] and historians [55] have used word clusters from topic models for corpus analysis. Nevertheless, successful application of topic modeling

⁴The system Themail [137] clusters words by time, instead of by co-occurrence statistics. Because this system shows lists of related words (related by time period), we say the system implements word clustering. Similarly, VisGets shows clusters (of document tags) defined by a user’s selection in the interface [41], which we consider to be a form of clustering.

requires specialized knowledge and extensive interpretive effort [5, 115], making this method less accessible to a broader audience of historians. Additionally, many historians approach archives looking for mentions of what Allen and Sieczkiewicz describe as “specific keywords” [2] rather than looking to explore word cluster overviews from a topic model API. Because we design for historians investigating known query terms (Section 1), we do not employ the word clustering pattern in the CLIOQUERY interface.

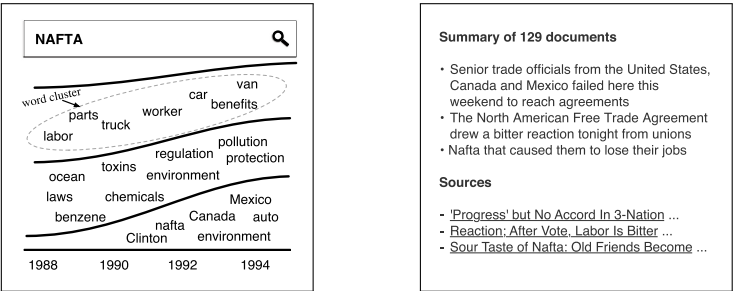
2.1.2 Textual and Visual Summaries. Rather than showing lists of related words to offer a corpus overview, a large body of work on text summarization from NLP [35] instead attempts to create short paragraphs that convey the most “important” information in a corpus by selecting a collection of sentences or sentence fragments from input documents to form an output summary. (This is sometimes described as extractive summarization [35], because the output text is extracted from input text.) User-facing systems such as Newsblaster [92] and NSTM [3] apply this research by showing such textual summaries in a graphical interface. We say that such tools implement the textual summary design pattern (Figure 3(b)). Other closely related work from text visualization considers how to present summary text in specialized visual layouts such as Document Cards [128], Phrase Nets [136], or Word Trees [139]. We say that these interfaces offer structured visual summaries, as they place summary text within some structured visual format (e.g., a directed graph [136]).

Like word clusters, both traditional text summaries and structured visual summaries do not seem to help with mention gathering and analysis. A user cannot turn to these forms of summaries to find and review query mentions, because “important” sentences selected for inclusion in summary output may or may not contain a given query word. Moreover, traditional approaches typically do not explain *how* “important” information is chosen, which may be important in the history domain (Section 9.3).

However, two ideas from the text summarization literature may help historians perform mention gathering and analysis. First, work in query-focused summarization tries to identify the most salient information in a corpus based on a user’s query [97]. Historians might use such query-focused summaries to review keywords in text. Query-focused summaries that define *all* query mentions as important enough to warrant inclusion in summary output may be especially helpful (see Section 3.3.2). Second, work on sentence compression [47, 49, 76] tries to shorten individual sentences by removing words, usually for the purpose of including more (shortened) sentences in a fixed-length summary. These methods, or closely related sentence fusion techniques [4], might be used to shorten passages containing query terms to help people quickly review many mentions of a query in context. We apply these two ideas from text summarization in CLIOQUERY (see Sections 4.3 and 4.7).

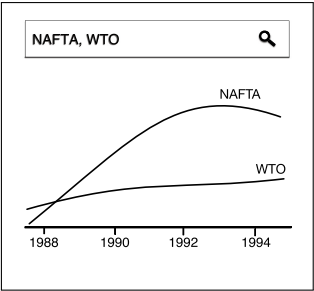
2.1.3 Time Series Plot. Instead of showing text to summarize corpus contents, time series plots present the frequency of words or documents across time to offer a visual (rather than textual) corpus overview. This pattern is often implemented in text analysis tools [36, 37, 86, 112] and keyword search systems [88, 100, 123]. Some time series visualizations [88, 93, 124] show the frequency of a single query term across time (e.g., Figure 3(c)), often using a line chart. Others show the frequency of multiple terms (e.g., highest-count words) using a stacked area chart [7, 61], and may not require a user-supplied query. While time series plots alone can not be used for mention gathering and analysis (because they do not show underlying text from a corpus), such visualizations can hint at important events or changes across documents (e.g., Michel et al. [93]). We thus implement this design pattern in CLIOQUERY (Section 4.2).

Overview
patterns
(Sec. 2.1)



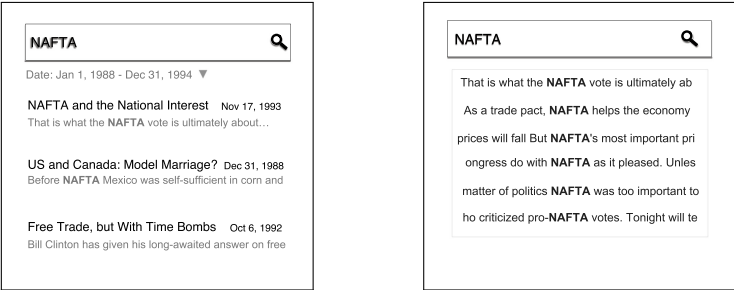
(a) Word clustering (Section 2.1.1) Examples: [13, 21, 31, 65, 100, 117]

(b) Textual & visual summary (Sec. 2.1.2) Examples: [3, 88, 89]



(c) Time series plot (Sec. 2.1.3) Examples: [7, 58, 90, 109]

Search
patterns
(Sec. 2.2)



(d) Keyword search (Sec. 2.2.1) Examples: [27, 59, 63, 96, 97, 109]

(e) Multi-doc. snippet (Sec. 2.2.2) Examples: [77, 101, 109, 134]

Fig. 3. We define five major user interface design patterns from prior work devoted to helping users understand news archives and other corpora (Section 2). Three of the design patterns focus on helping users gain an overview of archive contents (top two rows). Two of the design patterns focus on helping the user to search for specific documents or passages from a corpus (bottom row). Following Tidwell [132], this figure presents prototypical wireframes of each design pattern, created by the authors of this work. Each example above shows a system presenting results from 129 documents matching the query “NAFTA” on a corpus of *New York Times* editorials published between 1988 and 1994.

Table 2. Example Baseline Keyword Document Search Systems, Featuring Relevance-ranked Search Engine Results Pages and Filtering by Date

	Relevance ranking	Filter by Date	Known users
Chronicling America	✓	✓	<i>I3, I4, I5, P4, H1</i>
Newspapers.com	✓	✓	<i>I1, P1, P4, H2</i>
New York Times Search	✓	✓	<i>I3, I4, I5, P1, P5, H1, H2</i>
ProQuest	✓	✓	<i>I1 to I5, P1, P3, P4, P5, H1, H2</i>

Such features are common in many news archive interfaces [44]. Tables in the Appendix provide more details on the backgrounds of known users. Above, we use I1 to I5 to indicate all interviewees.

2.2 Search Design Patterns

2.2.1 Keyword Document Search (baseline). Traditional keyword document search tools return relevance-ranked lists of documents on a **search engine results page (SERP)** in response to a free-text query [84]. Because historians often use such tools in practice (Section 1), we consider these systems to be baselines for mention gathering and analysis.⁵

Although keyword document search tools are widely used (Table 2), these systems have clear downsides for finding and reviewing query mentions. First, keyword document search systems impose unnecessary burdens from reading and context switching. This is described in detail in Section 1. Additionally, keyword document search systems rank documents according to a computational model of relevance. This may be undesirable for historians, because relevance-ranking introduces opaque algorithmic influence over qualitative conclusions (by guiding people towards particular documents). Section 3 describes the importance of neutral and comprehensive review in historical research.

Ranking aside, keyword document search tools may also shape user perceptions of the contents of the individual documents in an archive through displaying single-document summaries (also called query-biased snippets [134]) on the search engine results page.⁶ For example, Figure 3(c) displays three sample single-document summaries, showing what a computer deems to be the most important information from three different search results. Such single-document summaries may be inappropriate for historical research, as some historians may be skeptical of opaque models that select “important” information for their review (search engines try to include keywords in snippets, but do not try to explain summaries [31, Section 6.3.1]). Prototypes shown in the Appendix describe our own experiences attempting to apply similar document summarization techniques for historians without success.

2.2.2 Multi-document Snippet. Where keyword document search systems return links to single documents in response to a user query, other systems return collections of smaller units such as paragraphs, sentences, or character spans, which are often drawn from multiple documents (see Figure 3(e)). We observe two different implementations of this multi-document snippet design pattern in interactive text analytics.

First, multi-document snippet features can be used in word clustering systems to help people investigate mentions of particular clustered words in context. For example, TIARA [78] allows analysts to review individual words from a cluster in underlying text. However, because TIARA is

⁵One strand of humanities scholarship critically investigates how widespread adoption of keyword document search tools might be distorting traditional humanistic research [110, 125, 135].

⁶Google sometimes shows complex results snippets on the SERP, using proprietary techniques. Brin and Page briefly mention the need for such “Result Summarization” in their original paper [15, Section 6.1].

designed for showing broad themes rather than for reviewing query mentions, it does not comprehensively show all mentions of a given word in its multi-document snippet. Instead, TIARA chooses some selection of mentions for display, optimizing for diversity [78, Section 6]. Such curation may introduce unwanted algorithmic bias (Section 3.3.4), because the system chooses some but not all query mentions for display.

Additionally, other text analysis systems that are not necessarily focused on clustering sometimes include **keyword-in-context (KWIC)** views [104, 112], showing each mention of a query word (or a selection of such mentions) on its own line of text amid immediately surrounding tokens or characters (e.g., Figure 3(e)). While this form of multi-document snippet can be used for mention gathering and analysis, KWIC views have some limitations for historical research. First, in many cases, historians need to investigate particular query mentions within the context of full documents (Section 3.3.3). While KWIC views may include links to underlying sources, jumping from KWIC views to documents requires context switching into new windows or tabs to gather and analyze evidence. We explain why this is undesirable in Section 1. Second, KWIC views always show some number of pixels, characters, or words immediately surrounding each query mention. This may result in awkward-sounding or choppy snippets that do not include the most salient information in source sentences; evidence suggests that people dislike awkward-sounding snippets [25]. Finally, KWIC views do not offer a way to keep track of which mentions have been reviewed during analysis, which may be important in historical research (Sections 3.3.2 and 4.5). Noting these shortcomings, it is possible to interpret certain CLIOQUERY features (Sections 4.3 and 4.4) as a particular form of KWIC view, addressing some of these limitations.

2.3 Situating CLIOQUERY within the Broader Literature on Visual Analytics for Text

Researchers have proposed many approaches to interactive text analytics [1, 12, 13, 23, 24, 33, 42, 53, 68, 71, 78, 82, 103, 120]. Within this broad literature, CLIOQUERY is unique because it is designed to help people find and analyze all occurrences of a query word across a corpus (see Section 3.1). Using terminology from Chuang et al., CLIOQUERY differs from prior work because its central “unit of analysis” [24] is the query word in context; the system’s “visual encodings,” “modeling decisions” [24], and user interactions were all designed to help people quickly and comprehensively review mentions of a query word across an archive. For instance, CLIOQUERY includes a textual summary feature that presents a synopsis of all occurrences of a query word in a corpus (Section 4.3).

Focusing on query words in context is a departure from prior work in text analytics, which emphasizes other latent and observable textual units of analysis, such as topics [78], events [82], document metadata [42], interrelated entities [53], or thematic hierarchies [13]. It is also a departure from keyword document search systems [99, 100], which focus on guiding people to ranked documents (Section 2.2.1).

Our atypical decision to design and build a text analytics system for analyzing occurrences of query words across a corpus followed from our systematic investigation into the tasks and expectations of historians and archivists (Section 3), who often review queries in text. Chuang et al.’s highly cited guidance for text analytics [24] stresses the importance of choosing units of analysis that are best “aligned” to the “tasks, expectations and background knowledge” of intended users.

Because CLIOQUERY uses query words in context as its central unit of analysis, the system differs from prior tools in several key ways. We highlight the most important differences below.

2.3.1 CLIOQUERY Displays Query Words in Underlying Text. Unlike CLIOQUERY, some prior systems for interactive text analysis are designed for analyzing high-level textual units such as corpus themes or temporal trends. As a result, these systems sometimes do not allow people to review

occurrences of query words in underlying documents. For instance, structured visual summaries such as the Word Tree [139] and Phrase Net visualizations [136], or time series displays like Theme River [61], do not show query words in underlying text. Similarly, some text analytics systems such as early versions of Overview [13] experiment with alternatives to query-based paradigms.⁷ By contrast, CLIOQUERY is designed to help people find and read query words in underlying documents; the tool’s central units of analysis (i.e., query words in context) are spans of text from documents in the corpus (see Section 3.1). This design choice was informed by prior work, which emphasizes the importance of displaying underlying text in interactive text analysis (see Section 9.2).

2.3.2 CLIOQUERY Offers Complete Access to All Query Mentions in Context, without Extraneous Mediating Abstractions. Like CLIOQUERY, some text analysis tools do include features to help people navigate to query words in context. However, because these systems are chiefly designed for analyzing other textual units (e.g., topics [78], events [82], or document hierarchies [13]), they offer only indirect and incomplete access to query words in underlying text, via extraneous mediating abstractions. By contrast, CLIOQUERY is designed to help people directly review all occurrences of a query in a corpus.

For instance, EventRiver [82] helps people review temporal document clusters, which serve as the system’s primary unit of analysis. In principle, a historian could use EventRiver to find and analyze query mentions in context by (1) finding document clusters containing a query word Q using the tool’s search-by-keyword feature, (2) clicking such clusters, and (3) using the tool’s Shoebox and Storyboard features to review those occurrences of Q that happen to fall within documents from selected clusters. However, such a workflow would have two downsides. First, the workflow would be *indirect*; the historian would have to navigate through clusters to access query words in context. Such indirect navigation would force the historian to attend to what Chuang et al. describe as “extraneous information that might confuse or hamper interpretation” [24]. Second, the workflow would be *incomplete*; a historian would have no way to navigate to occurrences of query words that do not happen to fall within algorithmically defined clusters. As we describe in Section 3, this would likely pose a problem for historians, who need to directly and comprehensively observe all mentions of their query in context, with minimal confounding algorithmic influence.

Our focus on EventRiver merely serves as one illustrative example of a broader phenomenon. TIARA’s mediating topic abstraction [78], Overview’s mediating hierarchy abstraction [13], Star-Spire’s mediating cluster abstraction [12], and Jigsaw’s mediating entity abstraction [53] (some queries are not entities, e.g., “race suicide” [83]) would also force historians and archivists to navigate to query mentions in context via similarly confounding and extraneous abstractions.

2.3.3 CLIOQUERY Employs Query-focused Summarization to Ease the Burden of Reading and Context Switching. Like CLIOQUERY, some text analytics systems include a snippet feature that shows words extracted from documents in a corpus. Examples include TIARA [78] Snippets, the Overview and Footprints Document List [13, 70], and results snippets from keyword document search systems [31, Chp. 6.3]. While such snippets are visually similar to CLIOQUERY’s Document Feed (Section 4.3), they differ in important ways that are crucial to the work of historians and archivists.

Most importantly, many existing snippet components select and display only some occurrences of a query in a corpus. For instance, TIARA’s Snippets feature displays a selection of occurrences of a topic word in underlying documents [78], and the Footprints [70] Document List displays the first sentence in a document (regardless of whether the sentence contains a query word). Similarly, keyword document search systems create and display snippets based on heterogeneous criteria [31, Chp. 6], rather than display all occurrences of a query word in ranked documents. Prior systems

⁷The Overview authors describe the importance of adding query features in discussing their work [13, 127].

also do not attempt to help people understand how text is selected for a snippet. For instance, the Overview Document List [13] selects and displays keywords based on an opaque clustering algorithm.

These properties make prior snippet features poorly suited to historians, who need to review all occurrences of a query word in a corpus with minimal algorithmic influence. Therefore, instead of relying on prior snippet features, a historian who wished to review all occurrences of a query word in a corpus would likely have to click through from snippets to underlying documents, which are often [53, 78, 99] shown in individual windows or tabs. In Section 1 and Figure 2, we explain how this workflow imposes unnecessary reading and context switching costs. By contrast, CLIOQUERY’s snippet-like Document Feed employs novel query-focused summarization techniques to allow historians and archivists to quickly scroll through and examine every single occurrence of a query term in a corpus. We offer qualitative and quantitative evidence of the importance of this feature in Sections 6, 7, and 8.

3 NEEDFINDING STUDY: DEFINING PRACTICES AND REQUIREMENTS

3.1 Formalizing Historians’ Current Practice of Mention Gathering and Analysis

In Section 1, we document and informally describe historians’ current practice of mention gathering and mention analysis. We now define this work more formally. During mention gathering, a historian investigates a unigram query Q in a newspaper archive. Q is a word type and each mention of the query, $i \in \mathcal{M}(Q)$, is a word token. For instance, a historian might investigate $Q = \text{“Falluja”}$ by gathering specific mentions of the word “Falluja” in individual documents, published on particular dates. Using d to refer to the text of a specific document and t to refer to its publication date, we can formally define mention gathering as the task of finding all $\mathcal{M}(Q)$ in an archive $\mathcal{A} = \{(d_1, t_1), (d_2, t_2) \dots (d_N, t_N)\}$, which is an unordered set of N timestamped documents.

The task of mention analysis consists of manually reviewing one or more query mentions in context. We use the notation $C(i)$ to refer to a specific passage showing a particular query mention in context, where $C(i)$ is a token span. For instance, if “Falluja” occurs in document d , then $C(i)$ might be a paragraph from d that contains the string “Falluja.” We denote this using $C_{\text{paragraph}}(i)$. Note that different systems may define $C(i)$ in different ways. For example, keyword document search systems return whole documents. Thus, a keyword document search system defines $C(i)$ as the whole document d containing $i \in \mathcal{M}(Q)$. We denote this using $C_{\text{full doc.}}(i)$.

Having now formally defined historians’ current practice of mention gathering and mention analysis and explained the limitations of baseline tools for these tasks (Section 1), we now describe an investigation into the needs of historians (Section 3.2), which informs our design requirements for a text analytics system (Section 3.3).

3.2 Observing and Analyzing Needs from Heterogeneous Data

We identified user needs by collecting and analyzing two different sources of data, described below.

3.2.1 Observing Needs from Existing Literature. First, we studied historians’ needs by reviewing a large literature from history, library science, and information science devoted to the systematic study of the digital and non-digital information-seeking behavior of historians. To identify this literature, we followed citations starting from Allen and Sieczkiewicz’s paper “How Historians Use Historical Newspapers” [2], which we first found via a search on Google Scholar. In total, we reviewed and took notes on six prior studies describing surveys and interviews with 1002 historians (shown in a table in the Appendix). We consider our synthesis of this prior literature to be part of the contribution of our work, as we translate these prior descriptive findings (focused on

how historians find information) into actionable design requirements for an interface. The studies we review are largely unknown in computer science disciplines such as NLP, IR, VIS, and HCI.

3.2.2 Observing Needs from Interviews and Feedback on Prototypes. We additionally supplemented, contextualized, and validated existing studies by conducting five of our own one-on-one needfinding interviews with five interviewees (I1 to I5) on Zoom video chat over a period of three months.⁸ The Appendix describes the backgrounds of interviewees in detail. All but one interview was 60 minutes long. (We met with I4 for 30 minutes, due to limited availability.) Interviews proceeded in two phases. During *Phase A*, in the initial exploratory stage of our work, one researcher from our group interviewed I2, I4, and I5, who we recruited through convenience sampling [51]. The interviewer asked open-ended, exploratory questions about needs and practices and solicited feedback on early prototypes. The researcher also took detailed notes. Later, when we better understood how historians find information in archives, we began *Phase B*. During this phase, the same researcher conducted two one-on-one, video-recorded, semi-structured interviews with I1 and I3, who also provided feedback on later prototypes. We recruited I1 and I3 via email outreach.⁹ The researcher again took detailed notes. We include the interview script in supplemental material. In total, each of the five interviewees across *Phase A* and *Phase B* reviewed a different iterative prototype. In the interest of space, we only present feedback on what we consider to be the two most important prototypes, shown in the Appendix.

3.2.3 Analyzing Observations of Historians' Needs. Following data collection, one researcher qualitatively analyzed and organized notes and transcripts to articulate four overall needs and translated these needs into four corresponding design requirements (described in Section 3.3). In general, we found that feedback from needfinding interviews and feedback on early prototypes was very consistent with findings from prior work. Nevertheless, our own needfinding interviews helped to contextualize and translate prior descriptive findings on historians' information-seeking behaviors into actionable guidelines for system design.

3.3 Needfinding Results and Design Requirements

Following data collection and data analysis, we defined four high-level design requirements (R1–R4), based on four needs. We describe each requirement below.

3.3.1 R1: A System Should Show a Navigable Overview of Change over Time. Prior study of the information-seeking behavior of historians emphasizes the theoretical importance of “*the dimension of time*” [19] in historical research and also emphasizes historians' practical need to perform “*searching and narrowing by date*” [2]. In our needfinding interviews, historians and archivists also stressed the theoretical and practical importance of time-based investigation. “*Time is always a historian's first move*,” I3 explained. “*It's about change over time as the fundamental thing*,” I5 noted: “*Historians are often trying to find articles within a specific date range and about a specific topic ... research often starts with a keyword and a date range and a source or list of sources*.” Because historical research involves studying change across time, I2 explained how time series plots showing the frequency of query words by time period (see Figure 3(d)) are often useful for gaining a temporal overview of a corpus. “*Bar charts [or line charts] by time are really helpful*,” I2 explained, “*because*

⁸Our needfinding interviews, expert interviews, and field study (Sections 3.2, 5, and 7, respectively) were approved as exempt from review by our institution's human subjects IRB office. All participants received a \$50 Amazon gift card for their time.

⁹We emailed five PhD students in history at a nearby university. Each student expressed interest in media, archives, or science in describing their work on their department's web page. We also emailed all members of the editorial board at a history journal. We do not list the name of the university or journal to ensure interviewees remain anonymous.

news has these peaks where a topic becomes important and then dies down.” Such charts “help people trace an idea or series of ideas or terminology over time.” Observing the centrality of temporal analysis in historical research, we assert a design requirement (R1): A system designed for historical mention gathering and analysis should show some kind of navigable, visual overview of query mentions $\mathcal{M}(Q)$ across the time span of a corpus. Showing such a visual “overview first” [63, 122] is a known best practice in visual analytics.

3.3.2 R2: A System Should Help People Comprehensively Review All Query Mentions in a Corpus. Prior work often emphasizes the importance of gathering comprehensive evidence during historical research. “Comprehensiveness is clearly the highest priority in searching a database,” one study concludes [34], explaining that 70% of 278 survey respondents would prefer to spend time filtering out irrelevant material than run the risk that relevant material “might fall through the cracks” in a limited search. Nevertheless, some historians in prior work acknowledge that truly comprehensive search is an impossible goal. “I never think I’m going to be able to read every record,” one reports [40]. “I’m always creating priority orders of what I think is going to be most useful.”

Our interviewees similarly emphasized the importance of comprehensiveness in gathering and evaluating historical evidence. “The most important thing for historical researchers is to be confident that they are being exhaustive,” said I4. “I want to know I can be confident I have been able to access everything relevant. Did my search cast a wide enough net?” I4 also praised an early prototype (see Appendix) for displaying a very large number of potentially relevant passages. “The biggest fear is Type II error,” he explained. “In doing searches, am I missing something that is crucial but I don’t know because I never looked?” Similarly, I5 explained that it is important to “be as completist as possible” in historical research. “The thing about historians....they want to be as comprehensive as possible with their topics.” Citing the importance of comprehensiveness, I5 expressed deep skepticism (see Appendix) about an early prototype that omitted some information to form a summary. However, like some interviewees in prior work, I2 pointed out that truly comprehensive investigation may not be possible. “Ultimately,” she noted, “there is a limit in terms of time and money for any given project.” We translate the need for comprehensive archival search into a second design requirement (R2): A system for mention gathering and analysis should help people comprehensively review all query mentions in a corpus. Expressed more formally, a system should help historians easily navigate to and review every single $i \in \mathcal{M}(Q)$ in an archive.

3.3.3 R3: A System Should Present as Much Context as Possible for Any Given Record in an Archive. Prior work emphasizes the importance of context in historical research. “Building context is the sine qua non [indispensable condition] of historical research,” Duff and Johnson write [40]. “Without it historians are unable to understand or interpret the events or activities they are examining.” In a separate study, another historian explains, “You can’t have the specific facts without the context ... Where an article is in the paper, and what surrounds it, matters.”

During our own needfinding interviews, historians and archivists also repeatedly emphasized the importance of contextual information in archive news search. The job of a historian is to “put facts in context,” I5 said. A historian will need to “contextualize” facts from a periodical by examining its publishers and audience. Similarly, I4 noted that “as an archivist I do research to give context to collections.” Finally, I2 stressed the importance of contextualizing evidence in archive search software. “Who does the New York Times have writing this?” I2 asked, while examining an early CLIOQUERY prototype. “Where does each sentence occur in the document? What section of the newspaper? You need to show more context.”

Observing the importance of context in historical research, we assert a design requirement (R3): A system for mention gathering and analysis should show each query mention amid as much surrounding context as possible. Formally, R3 implies that $C(i)$ should be as large as possible (in

token length) for each $i \in \mathcal{M}(Q)$. However, R3 must be balanced against other requirements, which impose competing demands. In particular, because screen space and human attention are limited resources, if $C(i)$ is large (e.g., a full document), then this will make it harder for a historian to comprehensively review all mentions in a corpus. Balancing a need for context and comprehensiveness is a challenge in designing for historians.

3.3.4 R4: A System Should Be as Transparent, Trustworthy, and Neutral as Possible. Prior studies of the information-seeking behavior of historians underscore the need for trustworthy tools that transparently present digital archival materials in a neutral manner. For instance, in one study [39], a historian reports that they prefer original sources because they can trust such sources to be “*accurate, undistorted and complete.*” Similarly, in another study [20], another historian explains that direct “*access to the original image of the primary source rather than to a transcribed version*” is important, “*especially when there is no description of what rules they used to transcribe documents.*” This historian reports that they do not trust and can not interpret electronic transcription, and thus must rely on direct observation of digitized images to draw conclusions.

In our interviews, historians and archivists similarly described the importance of transparently presenting digitized archives in a neutral manner. “*When I see something that is trying to decide or curate for me that is a worry. That is a red flag,*” I4 explained. Similarly, I2 added, “*I think the system should be as transparent as possible. I need to distinguish between what some primary source is saying versus what the computer thinks a primary source is saying.*” I5 also cited the importance of transparency and trust in expressing deep skepticism about an early prototype, shown in the Appendix.¹⁰ Because historians frequently expressed commitments to direct and neutral observation of archival evidence, we assert a design requirement (R4): Search software should show evidence in a maximally transparent and trustworthy manner. One consequence of R4 is that systems for mention gathering and analysis should not attempt to create a curated summary of the most “important” $\mathcal{M}(Q)$ in an archive (see Section 9.3).

4 SYSTEM

CLIOQUERY is a visual analytics system designed to support historians in their practice of mention gathering and analysis. The system is unique within a large literature on text analytics, because it is designed to help analyze query mentions in context, which constitute the system’s central “unit of analysis” (this terminology comes from Chuang et al. [24]). By contrast, prior text analytics tools focus on the investigation of other units of analysis such as topics [78], events [82], or thematic hierarchies [13] (see Section 2.3).

CLIOQUERY’s unique focus on query mentions in context reflects our study into the needs and practices of historical researchers, who helped refine early iterative prototypes [54]. Because such historians need to quickly and comprehensively review occurrences of a query word in a large corpus, CLIOQUERY uses text simplification techniques (Section 4.7) to create a skimmable summary of a query across an archive. Such text simplification techniques and skimmable summaries are unique within the large literature on text analytics and thus constitute the chief technical contribution of our work. We hypothesize that this query-focused summarization, along with CLIOQUERY’s linked views, in-text highlighting, and history tracking features, can help experts quickly, comprehensively, and transparently gather and analyze $\mathcal{M}(Q)$, the comprehensive set of all mentions of a query in a news archive.

¹⁰Even as some interviewees stressed the importance of unbiased, transparent, and trustworthy presentation of archive evidence, I3 reported that, in practice, historical researchers do trust ranked results from keyword document search systems. She explained that many historians might not realize that black-box document rankings from a keyword document search tool will affect conclusions from archival research.

4.1 High-level System Description

The CLIOQUERY web interface presents results from a Boolean search [84, Chapter 1], which returns the unranked set of documents containing one or more mentions of a unigram query term Q in an archive $\mathcal{A}$. (This notation and terminology is defined in Section 3.1; Section 10 discusses possible extensions to exact string matching.) When a user enters Q into the search bar at the top of the interface (Figure 1(A)), CLIOQUERY identifies all documents containing Q and presents the documents using three linked views [17]. First, CLIOQUERY includes a **Time Series View**, showing a graphical overview of the count of documents mentioning the query by year (Figure 1(D)). Second, CLIOQUERY includes a **Document Feed** view, presenting all query mentions from across all documents in a single scrollable window (Figure 1(H)). Finally, CLIOQUERY includes a **Document Viewer**, which shows the full text of a single document from the corpus, with individual query mentions from the document highlighted in context (Figure 1(I)). CLIOQUERY also includes a **filtering system** to help users narrow the set of query mentions shown in the interface (Figures 1(B), (C), and (F)) and a **history tracking system** to automatically monitor and display reading history during comprehensive search (Figure 1(G)). All features in the interface also follow a coordinated **color coding** scheme. For instance, the user's query word is always displayed using the purple query color ■ in the Document Feed and Document Viewer, and the Time Series View also uses a purple line to represent query frequency (Figure 1(D)). We consider the color-coded bolding of query terms to be one form of **automatic in-text highlighting** [57] throughout the CLIOQUERY interface. Automatic in-text highlighting draws user attention to some word, phrase, or passage in text by automatically setting the text's foreground color, background color, or decoration (e.g., bolding). The Appendix describes our process for selecting a colorblind-safe and print-friendly palette. It also provides additional engineering details about our implementation of CLIOQUERY.

4.2 Overview First: A Time Series View for Temporal Context (R1)

Because change across time is central to historical research (R1), CLIOQUERY presents a navigable Time Series View (Figure 1(D)) showing query frequency by year across a corpus. The component's x-axis represents time (binned by year) and its y-axis represents the annual count of all documents containing the query Q published during a given year. If a user also enters a subquery (Section 4.6), then CLIOQUERY's Time Series View also shows the annual count of documents mentioning both the query and subquery. In Figure 1(D), CLIOQUERY displays one line showing the count of documents mentioning the query term in the purple query color and another line showing the count of documents mentioning the subquery term (as well as the query term) in the green subquery color ■. CLIOQUERY's time series plot also shows a single rug point (small vertical line) for each document mentioning the query, just beneath the temporal x-axis (Figure 1(E)). Such rug points allow the user to easily preview and navigate to individual news stories; we describe these possible interactions in detail in the Appendix.

4.3 A Document Feed for Comprehensive Search (R2)

During needfinding, we found that experts often emphasized the importance of gathering comprehensive evidence (Section 3.3.2) and also often search for specific query terms in news archives (Section 1). We thus designed CLIOQUERY's Document Feed to help such users easily gather and analyze the comprehensive set of every single mention of a query term in a collection of news stories (R2). We assume the user is working with a small corpus (or small set of documents from a larger corpus), where such comprehensive review is possible. This assumption is appropriate for our use case; for instance, Black reviews roughly 500 documents to analyze the racial history of "watermelon," [9] and MacNamara reviews 605 documents to analyze "race suicide" [83].

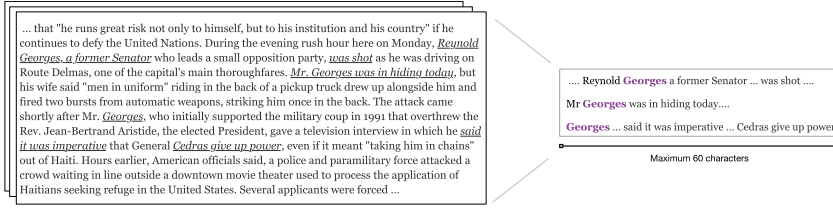


Fig. 4. CLIOQUERY's Document Feed uses text simplification methods from natural language processing in conjunction with color-coded, automatic, in-text highlighting to summarize query mentions $\mathcal{M}(Q)$ in context, in a visually consistent format designed for skimming. Here, one portion of one document (top of stack, left) containing three mentions of the query term $Q = \text{"Georges"}$ has been shortened into a summary of "Georges" (right). To create this summary, CLIOQUERY extracts each of the three sentences mentioning "Georges" and simplifies each sentence to render a shorter sentence $C_{s'}(i)$ that is fewer than 60 characters long. In this figure, for illustration, spans of tokens from one source document included in the summary are shown with italicized and underlined text. In typical use of CLIOQUERY there are often hundreds or thousands of documents containing Q (shown as document stack, above left).

After a user issues a query Q , CLIOQUERY populates the Document Feed to show a comprehensive, skimmable, summary that includes every single $i \in \mathcal{M}(Q)$ across the corpus (Figure 1(J)). To create the summary, CLIOQUERY selects each sentence containing some $i \in \mathcal{M}(Q)$ and then automatically simplifies the sentence (without removing query words) so historians can quickly read over the mention i in context. We use the notation $C_{s'}(i)$ to refer to a specific query mention $i \in \mathcal{M}(Q)$ shown within the context of a sentence s that is simplified to s' for display in the Document Feed. Section 4.7.1 provides details on how CLIOQUERY shortens sentences to create $C_{s'}(i)$. To the best of our knowledge, such text simplification is new to the literature on text analytics. In presenting each $C_{s'}(i)$, CLIOQUERY bolds and highlights the query word using the purple query color so each $C_{s'}(i)$ is shown in a visually consistent format designed for skimming (Figure 4).

Note that by default, CLIOQUERY displays a single $C_{s'}(i)$ from each document beneath the document's headline (the Appendix describes how the sentence is chosen). To see each $i \in \mathcal{M}(Q)$ from a document within a simplified sentence, the user can click an "expand" button (Figure 2). The user can also click a star to bookmark a document in the red bookmark color ■.

CLIOQUERY's Document Feed is designed to directly address two of the limitations of baseline keyword document search systems, described in Section 1. First, by summarizing documents mentioning Q , CLIOQUERY is able to fit more query mentions in limited screen space, reducing the need for context switching across individual windows or tabs. For instance, in Figure 1, the Document Feed saves the user from having to open 239 separate documents during comprehensive review. Second, by selecting sentences mentioning the query from documents and removing tokens from those sentences, CLIOQUERY reduces the user's reading burden. For instance, in Figure 1, the user has queried for documents mentioning Reagan and Duarte (in this example, Reagan is a subquery; subqueries are described in detail in Section 4.6). By selecting and simplifying sentences, CLIOQUERY removes 87.0% of the tokens in all documents mentioning these two words (Table 3). We include a detailed description of CLIOQUERY's text simplification techniques in Section 4.7.

4.4 A Linked Document Viewer for Necessary Context (R3, R4)

Because historians need to evaluate evidence in context without black-box algorithmic influence (R3, R4), we anticipated that CLIOQUERY users would need to quickly review each $i \in \mathcal{M}(Q)$ from the Document Feed within the context of full underlying news articles. Therefore, CLIOQUERY's Document Feed is closely linked with a corresponding **Document Viewer**, which shows the

Table 3. Total Tokens Presented to a Historian, if Each Mention of the Query “Reagan” and Subquery “Duarte” is Shown within the Context of a Full Document ($C_{\text{full doc.}}$), a Full Sentence ($C_{\text{full sent.}}$), or a Shortened Sentence ($C_{s'}$)

Context	Num. tokens
$C_{\text{full doc.}}$	222,544
$C_{\text{full sent.}}$	49,382
$C_{s'}$	28,859

By showing shortened sentences, CLIOQUERY removes 87.0% of tokens from all documents containing a query mention, and 41.6% of tokens from all sentences containing a query mention. In some cases, removing such tokens will exclude potentially relevant information from the summary; in these circumstances, a person would have to find and review such information using the Document Viewer. Counts come from the example in Figure 1, using the Salvador corpus (Appendix) and assuming CLIOQUERY is shown on a 13-inch screen.

complete text of a single selected document from the corpus (Figure 1(I)). The Document Viewer satisfies R3, because it shows each $i \in M(Q)$ within the context of a full document, denoted $C_{\text{full doc.}}(i)$. After a user clicks a shortened sentence $C_{s'}(i)$ in the Document Feed, the Document Viewer updates to show the entire document containing $C_{s'}(i)$. CLIOQUERY also automatically scrolls the document so the (just clicked) simplified sentence is visible on screen.

CLIOQUERY also makes it easy for users to locate simplified sentences by using automatic in-text highlighting to further link the Document Feed and Document Viewer. Each simplified sentence $C_{s'}(i)$ from the Document Feed is shown with yellow background highlighting  in the document shown in the Document Viewer. Additionally, if a user hovers over a sentence in the Document Feed or Document Viewer, then the sentence is highlighted in dark yellow hover color  in each component (shown in Figure 1(J) and 1(K)). We hypothesize that linking between shortened text and full documents helps build user trust (R4), because it helps experts transparently see and understand how shortened mentions are drawn from underlying text. This feature is inspired by CommunityClick [72].

4.5 Color-coded History Tracking for Systematic Review of Evidence (R2)

Some historical researchers emphasize the importance of comprehensively examining all available evidence during research (R2). To support historians in this work, CLIOQUERY keeps track of which documents the analyst clicks in the Document Feed and opens in the Document Viewer. CLIOQUERY also keeps track of bookmarked news stories (Figure 1(J)) and displays a simple stacked horizontal bar chart (Figure 1(G)) showing the proportions and counts of read, unread, and bookmarked documents. The bar chart uses the read , unread , and bookmarked  color scheme employed across the color-coordinated interface. (CLIOQUERY considers all documents to be either read but not bookmarked, unread or bookmarked. We do not allow intersection between these sets.) For instance, Figure 1(G) shows 5 read, 89 unread, and 5 bookmarked documents. The user can click check marks (Figure 1(G)) to show or hide documents in each category.

CLIOQUERY's Document Feed and Time Series View use the same color scheme to help users quickly identify opened and unopened documents. Stories that a user has already clicked appear with grey read text in the Document Feed, and their corresponding rug points are shown in grey

in the Time Series View. For instance, in Figure 1, the user has read the story published on Jan. 9, 1985. The story is greyed out in the Document Feed, and its corresponding rug point is shown in grey beneath the time series plot. Similarly, there are five red rug points in Figure 1(E), because the user has bookmarked five documents.

Note that CLIOQUERY's history tracking is query-dependent; tracking resets each time a user issues a new query (unlike the history-tracking mechanism in some prior work [70, Section 6]). Such query-dependent tracking is appropriate for CLIOQUERY, because the system is designed to help historians review all mentions of some specific keyword in a corpus. We hypothesize that this feature offers experts assurance they have comprehensively reviewed all $i \in \mathcal{M}(Q)$. We leave exploration of other forms of history tracking for future work.

4.6 A Filtering System to Review Many Results in a Neutral Manner (R4)

Some prior text analysis systems designed for historians (e.g., Expedition [123]) attempt to answer keyword queries by ranking documents to direct users towards most-relevant news articles. Because such ranked retrieval might introduce unwanted algorithmic influence over the expert search process (R4), CLIOQUERY responds to queries with Boolean search, which returns the unranked set of all documents containing Q . (The Document Feed shows such documents in chronological order.) CLIOQUERY then allows users to narrow down unranked search results with a filtering system, consisting of three filter controls.

The **filter-by-date** control selects documents by time period. After users select a start date and end date from date pickers at the top of the interface (Figure 1(B)), CLIOQUERY updates to show only those documents mentioning the query published during the selected interval. (Historians are often interested in specific time periods; see Section 3.) In Figure 1(B), the user has filtered to documents published in 1983–1985.

The **filter-by-subquery** control allows users to select documents that contain some additional word, called a subquery. For instance, after a user queries for the Salvadoran leader “Duarte” they might wish to further narrow results to understand the relationship between “Duarte” and his ally U.S. President Ronald Reagan. To investigate, the user can enter the subquery “Reagan” to select all documents mentioning the word “Duarte” that also mention the query word “Reagan” (Figure 1(C)). We included this feature because complex Boolean queries are often popular with experts [84, Section 1.4]. More complex Boolean expressions are possible in future work.

The **filter-by-count** control filters results based on the the number of times a query term is mentioned in a document. When a user adjusts the filter-by-count slider to some value $K \in \{1, 2, 3, 4, 5\}$ all components of the interface update to show only those documents with K or more $i \in \mathcal{M}(Q)$. In cases where a user has set a subquery, the filter-by-count control allows the user to select documents that contain the subquery word at least K times. For instance, in Figure 1(F), the user selects documents that mention “Reagan” at least three times.

CLIOQUERY also helps users quickly see the count of query terms within documents using square-shaped, query-colored **count markers**, shown beside each document headline. Count markers use brightness to encode the count of a query term within a document. For instance, count markers for documents with more mentions of a query term have a darker purple color than count markers for documents with fewer mentions. If a user enters a subquery, then count markers show the count of the subquery within each document, using shades of the subquery color (as in Figures 1(F) and 1(H)). This feature is inspired by TileBars [62].

4.7 Sentence Simplification to Help Summarize a Query across a Corpus

CLIOQUERY introduces text simplification methods from NLP to the literature on text analytics. We describe these methods below.

4.7.1 Overview of Sentence Simplification in CLIOQUERY. CLIOQUERY’s Document Feed displays a query-focused summary of a user’s query and subquery by first extracting and then simplifying sentences mentioning query (or subquery) words. To simplify sentences, we turn to sentence compression techniques from the text summarization literature in NLP (introduced in Section 2.1.2). These methods try to summarize naturally occurring input sentences by removing words to create shorter and well-formed output sentences that contain the most salient information from the input. (A well-formed sentence is one that sounds natural, rather than garbled or choppy [117].) In particular, we turn to a specific class of sentence compression methods, which can ensure that simplified sentences both (A) fit within limited screen space in a user interface and (B) mention the user’s query term or subquery term. Such methods are appropriate for CLIOQUERY, because each line in the Document Feed has a fixed width and must include some mention of the user’s query or subquery.

More concretely, we use a *query-focused clause deletion* [58, 60] method to shorten sentences in cases when a user has entered a query (Section 4.7.2) and also use *relationship span extraction* method [59] in cases when a user has entered both a query and subquery (Section 4.7.3). We also employ a final fallback approach, *character windowing*, when it is not possible to shorten a sentence using other techniques (Section 4.7.2). In the next sections, we describe each sentence-shortening method in greater detail. The Appendix provides additional details on how CLIOQUERY chooses between possible sentence shortening methods.¹¹

4.7.2 Query-focused Clause Deletion, and Character Windowing. CLIOQUERY’s Document Feed requires shortened sentences that mention Q and fit within available screen space. We assume that such shortenings should also be well-formed and contain the most salient information from longer source sentences. Prior research in IR suggests that users prefer well-formed snippets [25], and prior work in sentence compression [47, 48, 76] strives for both well-formedness and salience. We also assume that methods for constructing shortenings must run with low latency, which is known to be important in user-facing analytics systems [79]. Different sentence-shortening techniques might optimize for and manage tradeoffs between such requirements. But in this work, we turn to a simple *query-focused clause deletion* method to meet such criteria, allowing us to focus on how to apply text summarization methods in user interfaces for historical research.

Query-focused clause deletion exploits the fact that natural language sentences are sequences of words that exhibit hierarchical and nested grammatical structure [8]. For instance, the sequence “She swims in the pool” can be divided into interrelated word groups, with specific grammatical relationships; the words “in the pool” form a prepositional phrase that modifies the verb “swims.” To represent such linguistic structure, clause deletion employs a dependency parse tree [102] grammatical formalism. A dependency parse is a directed tree graph with one vertex for each word in the sentence, along with a latent root vertex.¹² Each subtree in the parse corresponds to a constituent subsequence in the sentence. The sentence simplification literature sometimes describes such subtrees as *clauses* [49]. Figure 5(a) shows an example dependency parse.

Sentence simplification via clause deletion shortens sentences by iteratively deleting clauses from a dependency parse.¹³ Figure 5 shows how one sentence is shortened by iteratively deleting

¹¹In Figure 1, CLIOQUERY uses relationship span extraction to shorten and display some sentence from 31 out of 239 documents that mention “Duarte” and “Reagan.” It uses query-focused clause deletion to shorten and display some sentence from 85 documents, and it resorts to character windowing for the remaining 123 documents.

¹²We use the UD (v1) dependency formalism [102]; other related formalisms allow for non-tree parses [116]. Eisenstein [45, Chapter 11] offers a broad introduction to dependencies. We perform dependency parsing using Stanford CoreNLP [21, 85].

¹³Tokens from the remaining tree are then printed in left-to-right order, based on their position in the original sentence.

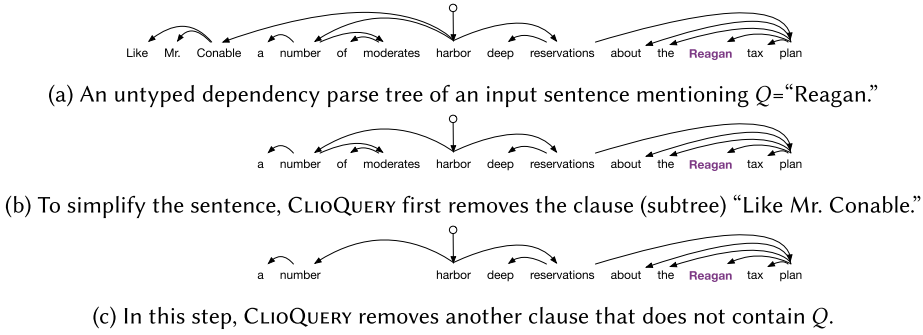


Fig. 5. Sentence simplification via query-focused clause deletion [58, 60]. CLIOQUERY removes two subtrees from a dependency parse across two steps to simplify the input sentence (5(a)) into the output sentence (5(c)).

two clauses. Unlike sentence compression techniques that consider individual tokens for removal (e.g., Filippova et al. [47]), deleting clauses naturally identifies and removes groups of related words. For example, a single deletion could remove the prepositional phrase "after the election," or a much longer word group with more modifiers and embedded clauses: "after the previous election last year, which went poorly." Shortening sentences via clause deletion also makes it easy to ensure that output sentences must include Q ; clauses that contain query mentions are not allowed to be removed during deletion.¹⁴

To try and create well-formed output sentences, CLIOQUERY turns to prior work on clause deletion [58, Section 6], which has found that in general removing more clauses from an input sentence makes it less likely that the resulting output sentence will be well-formed. Thus, to shorten an input sentence, CLIOQUERY's clause deletion first identifies those candidate output shortenings that can be constructed by removing at most K clauses from the input (without removing Q) and are also short enough to fit in one line of text within the Document Feed. Because in practice it is often possible to dramatically shorten an English news sentence by removing only one or two large clauses (for example, a lengthy relative clause, such as "Reagan met with the envoy ~~who was sent by the ...~~"), CLIOQUERY only considers shortenings that can be constructed by removing $0 < K \leq 2$ deletions.¹⁵

To try and ensure that output shortenings include the most salient information from input sentences, CLIOQUERY then returns the candidate output shortening with the highest tf-idf score [84]. Tf-idf scores are often used in extractive sentence compression [26, 49] and text summarization [35] to identify salient information for inclusion in summary output; this metric identifies words that occur with unusual frequency (relative to the overall corpus), which is an important signal

¹⁴It is also possible to enforce such query constraints using integer linear programming (ILP). However, ILP-based sentence compression techniques (e.g., Clarke and Lapata [26]) are NP-hard and have been shown to be orders of magnitude slower than other iterative approaches to query-focused sentence compression [60].

¹⁵In addition to encouraging well-formed output, this strict limit ensures low latency for the user. For a sentence M words long, the worst case for performance is a tree where all words are leaf vertexes, resulting in $M + M(M - 1)/2$ possible outputs of $K = 1$ or 2 deletions. But in typical trees, there are far fewer possible deletions because: (1) the query word and all its ancestors are not allowed to be deleted, (2) after the first deletion of a clause length C (i.e., the size of the deleted subtree) only $M - C$ candidates remain for the second deletion, and (3) if CLIOQUERY finds any candidate shortenings using $K=1$, then it will not search for candidates using $K=2$, as shortenings that remove fewer clauses are more likely to be well-formed. We do not consider cases where $K = 0$, as most unshortened news sentences are too long to fit within the Document Viewer.

of salience in summarization [98]. The Appendix includes details of how we compute *td-idf* in CLIOQUERY to identify words that occur frequently in documents mentioning a query.

In some cases, there is no way to shorten a sentence by removing one or two clauses while ensuring that that output sentence mentions Q and will fit in the Document Feed. In these circumstances, CLIOQUERY resorts to shortening the sentence by extracting the span of N characters to the left and right of Q in the sentence, where we maximize N under the constraint that the resulting character span will both fit in Document Feed and respect word boundaries. We use this *character windowing* method only as a last resort, because it may cut off syntactic constituents (e.g., show only a portion of a prepositional phrase), which may create awkward-sounding output. Footnote 11 describes how often CLIOQUERY uses this fallback during an example run of CLIOQUERY.

In the future, it might be possible to shorten more sentences with query-focused clause deletion by considering candidate output shortenings that are created using more than $K = 2$ deletions. (Prior work on query-focused clause deletion does not yet offer an efficient solution for considering such candidates [58].) Because the number of candidates grows with K , developing algorithms that efficiently search over possible outputs or learn greedy deletion policies based on data (e.g., with reinforcement learning) might offer useful starting points.

4.7.3 Relationship Span Extraction. CLIOQUERY users who search for a query term Q can also filter query results by a subquery. When a user enters both a query and a subquery term, we assume that they are broadly interested in how these two terms are related in the corpus. For instance, a user might query for the Salvadoran leader $Q = \text{“Duarte”}$ and apply a subquery for the then U.S. President “Reagan” to understand Duarte’s relationship to Reagan (Figure 1).

To meet this information need, CLIOQUERY attempts to simplify long and complex sentences mentioning both the query and subquery terms into short sentences that concisely describe the relationship between the query and subquery. We describe the process of shortening sentences in this manner as *relationship span extraction*, because each shortened sentence is a token span (i.e., sequence of tokens) extracted from a longer sentence. For instance, in Figure 1, we extract the span “Reagan sent congratulations to Mr. Duarte” from the longer sentence “President Reagan sent congratulations to Mr. Duarte and Ambassador Thomas R. Pickering pledged United States support for further meetings.”

CLIOQUERY relies on a known natural language processing technique to perform relationship span extraction, which is specified in detail in prior literature [59, Section 4]. At a high level, this method employs logistic regression to determine if an input sentence s containing two input query words can be shortened to express a relationship between those two query words. To make this determination, the method first extracts a vector of linguistic features $\mathbf{x}$ containing information about the query words in the sentence (e.g., is there a verb token between the query words in s ?), and then passes the dot product of $\mathbf{x}$ and a learned weight vector θ through a logistic function σ . This returns a predicted probability that the token span between the query and subquery will sound natural when removed from the sentence (Figure 6). In our case, the input query words are the user’s query and subquery; we shorten a sentence s to a relationship span if the predicted probability that the span sounds natural is greater than a threshold $T = 0.5$.¹⁶

In our implementation of CLIOQUERY (as in prior literature [59]), relationship span extraction is supervised using a benchmark corpus from Filippova and Altun [48], consisting of single sentences paired with single-sentence summaries, which are automatically generated from news headlines. In principle, a technically oriented CLIOQUERY user would be able to retrain relationship span

¹⁶We implement with Scikit-learn [105]. Note that setting a lower threshold T might increase the total number of shortened sentences at the cost of creating fewer well-formed extractions (and vice versa).

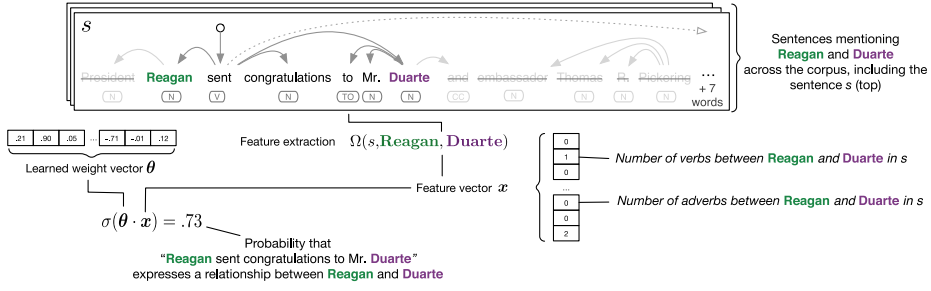


Fig. 6. Sentence simplification via relationship span extraction [59]. This method predicts the probability that the span of tokens between the query and subquery words in a given sentence s will sound natural as a shorter and standalone sentence. (If the predicted probability is high, then the sentence can likely be shortened to the span of tokens.) This figure shows the predicted probability that the token span between the query **Duarte** and the subquery **Reagan** will sound natural as a shortened sentence when extracted from the longer sentence shown in Figure 1 (letter K). Seven words from s are not shown in the diagram above.

extraction on their own corpus, using the technique from Filippova and Altun to automatically generate training data from headlines in their own news archive.

5 EXPERT INTERVIEW STUDY PROCEDURE

After analyzing user needs (Section 3) and designing the CLIOQUERY system based on such needs (Section 4), we conducted an interview study over Zoom video chat to test CLIOQUERY with historians and archivists who used the system to investigate questions from news archives.

5.1 Recruitment, Participants, and Corpora

We recruited five participants (P1–P5) from two universities in the U.S. by emailing students, faculty, and staff listed on history and library department web pages. All participants had advanced degrees (master’s or PhD) in history or library science, much like the expected users of our system. We provide more details on the backgrounds of participants in the Appendix. Interviewees from our needfinding study (Section 3) did not participate in our expert interview study to avoid what Sedlmair et al. describe as a potential form of bias [119]. Each participant in the interview study had an established research or curatorial interest in some topic related to late 20th century or early 21st century history, which we express as a single topic word (see Appendix). We identified this designated topic word based on each participant’s publication record and professional web presence. Before each interview, we then loaded CLIOQUERY with a corpus of *New York Times* (NYT) editorials¹⁷ published between 1987–2007 [114] mentioning the designated topic word.

5.2 Data Collection

To administer the study, one researcher from our group conducted five, one-on-one, 60-minute interviews over Zoom video chat. (See supplemental materials for a detailed script.) During each interview, the researcher asked each participant to brainstorm and then articulate a high-level research question based on the participant’s prior work (10 minutes). They then introduced the participant to CLIOQUERY via a tutorial (7 minutes), and asked them to investigate their research question using CLIOQUERY (30 minutes). They concluded with a semi-structured interview

¹⁷Social researchers sometimes study editorials to better understand media sources [22, 81].

(13 minutes). Throughout, the researcher observed and recorded participant reactions and invited participants to think aloud [101] as they used the system. If a participant offered feedback on some portion of the interface during their investigation (e.g., offered detailed feedback on the Time Series View), then the researcher did not ask about this topic again during the semi-structured interview.

5.3 Thematic Coding

The researcher who conducted the interviews analyzed automatic Zoom transcriptions for each of the video recordings and corrected transcription errors. The researcher then extracted 183 quotes from across five interview transcripts. Each quote consisted of a few sentences on a focused topic, along with the preceding question or comment to provide context (e.g., a quote might discuss the Document Feed). The researcher attempted to extract as many quotes as possible, while excluding irrelevant quotes (e.g., tutorial instructions).

The researcher then developed a codebook of six high-level codes (described in Section 6) by grouping and re-grouping the 183 quotes to identify common themes, much like the codebook-based approach described in Miles, Huberman, and Saldaña [95, Chp. 4].¹⁸ After each assigning each quote to exactly one of the six codes, the researcher shared the codebook with an undergraduate coder with training and experience in qualitative coding (who was not involved with the development of CLIOQUERY). The second coder independently assigned codes to the same quotes, using the codebook. The second coder was also invited to add new codes to the codebook if needed, but reported that no new codes were necessary. (Thus, we did not modify the codebook.) We include a copy of the codebook in supplemental materials.

Following independent coding of each of the 183 quotes, the two coders met for one hour over Zoom video chat to discuss 41 disagreements and attempted to reach consensus via discussion. In 21 cases, the two coders were able to reach agreement regarding the appropriate code. In 20 cases, the coders determined that disagreement reflected genuine ambiguity in qualitative data and agreed to disagree.

McDonald et al. [89, Section 2.2] use the term *reliability* to describe the extent to which coders reach the same result from independent work and use the term *agreement* to describe the extent to which coders reach consensus after discussion. Adopting this terminology, we measure the reliability of the two coders by computing a Cohen's $\kappa = 0.724$ (using the R psych 2.0 package [111]), and we measure the agreement of the two coders by computing a $\kappa = 0.855$.

6 EXPERT INTERVIEW STUDY RESULTS

Six themes emerged from qualitative coding, described below.

6.1 CLIOQUERY Helps with Historical Sensemaking

While using CLIOQUERY, each of the five participants formed a question and then collected and interpreted evidence to start to answer that question. We observed that CLIOQUERY helped historians with this investigative process, which Dalton and Charnigo [34] describe as historical sensemaking. Our observations offered partial validation for our hypothesis that CLIOQUERY features can aid historians in their work (Section 4).

For instance, as part of his research, P1 studies *New York Times* news coverage from journalists embedded with United States military units in the Iraqi city of Falluja during the second U.S.–Iraq war. From prior study, P1 understood that embedded U.S. journalists often published news stories reflecting the perspectives of U.S. military leaders. But while examining mentions of *Q* = “Falluja” in *New York Times* editorials using the CLIOQUERY interface, P1 expressed surprise when

¹⁸Miles, Huberman, and Saldaña [95, Chp. 4] describe assigning codes in two phases; we assign codes in a single phase.

finding a more nuanced perspective from the opinion desk. *"I didn't see nearly as much of the sort of sensational depiction of Falluja, and the militants in Falluja [in editorials] that I expect from embedded journalists [in news stories],"* he reported.

Similarly, P2 used CLIOQUERY to find confirming evidence of shifting U.S. perspectives towards Robert Mugabe. As P2 expected, early *New York Times* editorials from the corpus praised Q = "Mugabe" as a liberator, but then began to criticize *"him as a bad statesman, as a tyrant and a dictator."* P3 was likewise able to partially answer a research question with CLIOQUERY. She explained that while she had *"a deep knowledge of [women in combat]. I don't have a deep knowledge of what the [NYT] editorial board has to say about it."* Using CLIOQUERY, she found evidence of editorials using *"the gendered trope that women are supposed to be wives and mothers."* P5 also discovered an unexpected connection with musical copyright, while researching a hypothesis surrounding literary copyright. *"The parity [with the music service] Napster that's really interesting ... That's not something I thought about ... I was thinking ... definitely more in literary items because that's what I deal with."*

6.2 CLIOQUERY Features Offer a Corpus Overview, alongside Complementary Context

Participants offered detailed feedback on CLIOQUERY features during interviews, which often matched our design goals for particular components of the interface. To begin, three participants reported that CLIOQUERY's **Time Series View** offered a useful overview of the entire corpus by directing their attention to salient time periods. P1 said the Time Series View was an *"easy way of visualizing"* corpus trends, and P5 suggested that the Time Series View might be helpful *"when students are kind of in that exploratory phase ... as a way of ... coming up with research questions."* P4 offered similar feedback. *"I really like this,"* she said. *"This looks really functional and really useful. I like how there is quite a lot of information packed in."*

P1, P2, and P5 reported that the **Document Feed** was a useful feature of CLIOQUERY, because it helped summarize query mentions. The Document Feed *"condenses all of the essential information and sort of leaves out all the extra stuff,"* said P1. Said P2, *"I found [the Document Feed] useful, especially the expand button. If I click expand I can see a rundown of the mentions right after the title without seeing the article."* P5 reported using the Document Feed to *"do some ... simple kind of topic modeling in my own head ... just to see if I could pull out any ... themes there."* P5 added that, *"having this here [i.e., Document Feed] is really helpful to kind of see what they're talking about."* P3 and P4 discussed the Document Feed while describing the importance of context in historical research; we include their feedback on this feature in Section 6.4.

Several participants also reported that the **Document Viewer** helped during their research. For instance, P3 reported that automatic in-text highlighting in the feed was very helpful. *"I'm a visual person. So I'm looking for the words. I like that they're in purple and green ... the words that you've given me the pop out ... and I can see if it's a pro or con article pretty quickly just from that."* P2 said he used the Document Viewer to *"provide detail."*

P1, P2, and P5 noted that CLIOQUERY's linked **Document Feed and Document Viewer** served complementary purposes. They described how the Document Feed provided a summary of the query term, while the Document Viewer provided necessary and complementary details. *"You need both [the Document Feed and Viewer],"* said P2. *"With just the Document Feed I won't be able to get the full picture of the story. And with just the Document Viewer I will not be able to trace the mentions quite comprehensively and specifically."* P2 then added, *"as a researcher, it's important to see things in detail. If you just conclude from what you see in the Document Feed you are not going to get an objective picture of the context of the story line. But if you see the Document Feed, see the mentions, see what they imply, and then you want to understand the context of the story you are going to get to the Document Viewer."* P1 said, *"I like having both the Document Feed and the Document Viewer side*

by side. [The Document Viewer helps with] reading for more depth when I want more depth and [the Document Feed] helps with ... quick scans pretty easy.” Similarly, P5 explained, “I see [the Document Feed and Viewer] working together really well ... I start by looking at the feed to kind of pick out the articles that would want to kind of dive into deeper and then I go into the Document Viewer.”

P4 and P5 specifically mentioned that complementary linked views from the Document Feed and Document Viewer helped with **mention gathering and analysis**, as compared to a baseline keyword document search system. “A lot of databases that we work with do something similar to this [i.e., the Document Feed],” said P5, while describing a search engine results page. “But you often then have to click on the article to go into the article to get to that reading ... here it is nice that it was just kind of next to it and you can scroll through it.” Similarly, P4 described the difficulties of context switching between documents from the Google search engine results page. “Obviously, it’s a time saver,” she said, comparing CLIOQUERY to the keyword document search system. “You can tell ... just using the editorials at one newspaper.”

Two participants relied on CLIOQUERY’s **filtering system** to investigate their research topics. P1 investigated the NYT editorial board’s discussion of the query term “Falluja” using the filter-by-subquery feature (e.g., searching for “Falluja” and “resistance” or “Falluja” and “terrorist”). “It’s pretty interesting to me that I get three hits with the words Falluja and resistance and only one with the word terrorist,” he said. “That would suggest a certain orientation from the editorial board that will be unexpected.” P2 found the filter-by-count feature very helpful. “Oh, this is good,” he said, while testing out the slider. “It gets us through to the most important, the most critical pieces that we want to read.”

6.3 Some Disavow Obligation to Perform Comprehensive Review, Noting High Costs

During needfinding, interviewees emphasized the importance of comprehensively reviewing all available evidence. However, to our surprise, during the expert interview study, P4 explicitly disavowed an obligation to search comprehensively. “I don’t feel like I have an obligation to look at everything,” she said. “I have an obligation to get an overview and I think you know, with a completely unscientific measure of, oh, I think I’ve got enough now.” Similarly, P1 commented that, “I don’t think anyone actually does it [search comprehensively].” He went on, “A lot of people pretend they do it ... [but] in terms of like visiting archives ... everyone’s skimming ... they already know what they’re looking for and they’re just trying to find it.” P2 pointed out that comprehensive manual review was desirable but ultimately had high costs. “I am not saying we should get rid of personal scrutiny, the way you do it yourself. [But] you want to save time. If you do it [i.e., read] one-by-one it wastes too much time.” We discuss ambiguity surrounding comprehensive review in Section 9.

6.4 Context Is Crucial in Historical Research, so Some Are Wary of Text Summarization

Like during needfinding (Section 3.3.3), participants often emphasized the importance of context in historical research. For instance, P3 described extensive research to prepare for oral history interviews to “get that context to be able to ask them the questions that I asked them.” P2 also reported that context is “very important” for historians, as it “helps you understand why things are what they are.”

Some historians’ emphasis on context informed their feedback on the Document Feed. While P1, P2, and P5 found the Document Feed useful (Section 6.2), P3 and P4 expressed reservations, because they felt they needed more context to reach conclusions. P3 took the more extreme position. “For me, I don’t know if [the Document Feed] is necessary,” she said. “As a history scholar, you can’t take things out of context. You need to know the bigger context.” However, P4 reported that she would need more context (i.e., longer extractions from news stories) before the feature would be useful.

“The more context I can take in within as compact a time frame and compact a format, but sufficiently informative [the better]” she said. “But I think these [shortened sentences in the Document Feed] might have to be longer for that to work.”

6.5 Some Users Recognize a Tradeoff between Neutral Review and Limited Time

During needfinding and prototyping, interviewees often stressed the importance of avoiding possible bias from software in historical research. But during our expert interview study, P4 reported that she relied on black-box relevance models to direct her attention while searching archives. *“I do try to use the chronological sorting [when using ProQuest],”* said P4. *“But it is ... too much to wade through. If your corpus is reasonably big then you have to have a relevance kind of algorithm in there. Otherwise, it’s just going to be too frustrating.”* P4 also recognized that reliance on ranking introduces confounds. *“I think it would be appropriate to make people look at all of the irrelevant stuff,”* she said. *“So they realize the algorithm is pulling the relevant stuff for you ... but you can’t make the search s*** for people just to sort of make that point.”*

However, P5 liked how CLIOQUERY used filtering to avoid potential bias. *“I think it’s better that its just showing everything,”* he explained. *“I prefer having everything there to kind of whittle down ... as opposed to having certain things like cherry-picked ... I guess it’s never super clear to me why certain things might be moved to the top of results ... it raises questions about how things are ordered and how they’re brought to light.”*

As I3 predicted (Section 3.3.4), P1 described relying on the search function of the *New York Times* website [99] without understanding how the site was ranking search results by relevance. *“I wasn’t super aware of how they were pulling up articles for me ... They rank it in terms of views right?”* he said. He added, *“I just don’t, you know, have the knowledge of how to navigate these ... search engines well enough.”* We discuss mixed feedback on algorithmic bias in Section 9.4.

6.6 Access, Integrity, and Integration Are Important to Current Practices

Many participants commented on the importance of access, integrity, and integration in describing their current practices with newspaper archives (see also Section 10). P1 reported gathering news articles on U.S.–Iraqi relations from around the web “for years” by using search engines like Google or the *New York Times* website [99], saving these articles to the Internet Archive [28], and then organizing this collection using the software program Omeka [30]. This participant pointed out that CLIOQUERY *“assumes you have found all the stuff you want to work with,”* which is not true for his current research. P2 said that he had to rely on physical archives of print newspapers in Zimbabwe, which required burdensome international travel. P3 said that she rarely used newspapers in her own research, because many newspaper archives are often inaccessible behind paywalls, and P4 emphasized the need for better optical character recognition technology to improve search over printed newspapers. P5 reported that he *“used Zotero a lot”* to store and organize archival sources; he liked that Zotero is open source and integrates with Microsoft Word.

7 FIELD STUDY

In their review of design study methodology, Sedlmair et al. emphasize the importance of deploying a designed solution “in the wild” to test if new software helps “real users” solve “real problems” with “real data” [119]. Thus, we deployed CLIOQUERY over the web in a field study for two historians who used the tool to answer questions from their own research. Unlike in the expert interview study, during the field study, historians investigated questions over multiple meetings, and tried to reach substantive rather than preliminary conclusions. We believe that this evaluation offers more realistic but also less uniform feedback than the one-hour expert interviews described in Section 5.

7.1 Procedure

We recruited two historians, *H1* and *H2*, through convenience sampling [51]. The Appendix includes details on their backgrounds. *H1* and *H2* did not participate in the initial design or development of CLIOQUERY to avoid what Sedlmair et al. [119] describe as a potential source of bias. During the field study, one member of our research team conducted three one-on-one meetings with each historian over Zoom video chat. The first meeting was 30 minutes long and the subsequent meetings were 60 to 70 minutes long, with one to three weeks between each meeting. Each meeting in the three-meeting sequence had a distinct focus. During the first meeting, the researcher presented a tutorial of the software, described the field study process, and invited the historian to describe a question related to their research. After the first meeting, a member of our research team gathered the data needed to answer the historian's research question and loaded it into CLIOQUERY (the Appendix describes this data gathering). During the second meeting, each historian learned to use the CLIOQUERY software and performed a preliminary exploration of the data. Then, during the final meeting, each historian investigated some specific query by analyzing the comprehensive set of all mentions using the Document Feed and Document Viewer. During each meeting, the researcher observed each historian and invited the historian to think aloud [101] as they used the system. The researcher also asked the historian to describe their findings and explain how CLIOQUERY helped or did not help answer their research question.

7.2 CLIOQUERY Helps Experts Investigate by Skimming, an Advantage over Baselines

During the field study, *H1* and *H2* each used CLIOQUERY to reach substantive historical conclusions, offering additional evidence for our hypothesis (Section 4) that CLIOQUERY can help experts answer research questions from news archives.

H1 used CLIOQUERY to verify a well-known claim from Herman and Chomsky, who argue that for-profit news organizations in the United States shape public opinion towards the interests of political and economic elites [64]. To offer evidence for this theory, in their work, Herman and Chomsky assert that *The New York Times* wrote five articles in February and March of 1984 describing the Salvadoran army as a protector of El Salvador's election. To verify this result, *H1* searched a *New York Times* corpus (see Appendix) for the query "election" and then used the filter-by-date feature to select articles from February and March of 1984. *H1* then used the filter-by-subquery feature to identify those query results that contained the subquery "army." *H1* then systematically reviewed all 32 matching documents, through what *H1* described as "*skimming highlighted parts*" in the Document Viewer. By using CLIOQUERY in this manner, *H1* said that they were "*able to find what might be the five articles*" Herman and Chomsky used to partially support their conclusions. *H1* explained, "*The tool is great for exactly this.*"

H1 found CLIOQUERY's in-text highlighting helpful for their research task, drawing a comparison with a baseline keyword document search system (Section 2.2.1). "*I like how you have the bold highlighted and colored words in the text itself,*" they said. "*That is the advantage that this interface has over the New York Times website.*" *H1* also explained how such highlighting reduced reading burden (compared to a keyword document search). "*What I need to know is the army described as a protector of the election [in an article],*" he said. "*I don't need to read every word of the article to find that out. I can look at the paragraphs where they are describing the army and I see what they are saying in those paragraphs. That is pretty useful.*"

H2 chose to use CLIOQUERY to study how the United States media represented female astronauts Svetlana Savitskaya and Sally Ride in the early 1980s. (*H2* needed to answer this question to research a planned book.) To investigate, *H2* used CLIOQUERY's Document Feed and Document Viewer to review portrayal of Sally Ride in *The New York Times*. *H2* queried for the word "Ride"

and then scrolled through the Document Feed to skim over mentions of Ride in the 63 matching documents, sometimes also clicking to open individual news stories in the Document Viewer. “I have some hypotheses that I was able to develop very quickly through the experience of using this [system],” H2 reported. “One is that Ride was presented to the American public [in *The New York Times*] ... first as a woman and second as a scientist.” H2 asked us to continue to provide access after the study, so she could continue researching her book using the tool.

CLIOQUERY’s Document Feed was particularly helpful for H2, who found that query-focused summarization offered an advantage over a baseline keyword document search system. Ride was a PhD astrophysicist-turned-astronaut, and H2 wanted to understand how the media portrayed her scientific credentials. The Document Feed helped H2 quickly review this information. “[Here] she’s called a flight engineer,” H2 said, pointing to the Document Feed. “I can see this already [without opening the document].” H2 then scrolled through the Document Feed to find shortened sentences where Sally Ride was described with her academic title (Dr. Ride) and sentences where Ride was described (or not described) as a physicist. H2 explained that she could identify this information “just doing the quick scan [in the Document Feed].” She went on to explain how she would normally research this question with *The New York Times* archive (by opening and reading individual news stories using a web browser). “The question is,” she said, “what can I do here [with CLIOQUERY] that I can’t do there [i.e., on *The New York Times* website]?” H2 continued, “It’s exploring the left hand Document Feed here. This is awesome ... I am liking these short contextual pieces [i.e., shortened sentences].” We illustrate this comparison in Figure 2; by using CLIOQUERY, H2 was able to easily gather and analyze mentions of Ride across the corpus.

8 A QUANTITATIVE COMPARISON WITH KEYWORD DOCUMENT SEARCH TOOLS

8.1 A Crowdsourced Historical Reading Comprehension Task

We designed a crowdsourced historical reading comprehension task to compare CLIOQUERY with a keyword document search system (IR), which we consider to be a baseline tool for historical research (Sections 1 and 2). Our task is designed to reflect historians’ common practice of mention gathering and analysis, in which expert social researchers find and review occurrences of a query Q in an archive (Section 1) to draw conclusions about society. In our crowdsourced adaptation of this common historical research process, we tasked non-specialists with finding and reviewing occurrences of a query in a newspaper corpus.¹⁹ We then used reading comprehension questions to measure how well participants performed at finding and reviewing information about the query; many common educational assessments use similar reading comprehension questions to assess how well people learn information from documents [6, Chp. 7].

To ensure we presented an ecologically valid research prompt, we modeled our crowd task after a real historical question from P2. In our interview study (Section 5), P2 used CLIOQUERY to investigate if *The New York Times* portrayed the controversial figure Robert Mugabe as a corrupt authoritarian, or as a hero of Zimbabwe’s fight for independence. In our crowd study, we presented participants with one of two text analytics tools loaded with the same small corpus of 12 *New York Times* editorials mentioning Robert Mugabe, published from January, 2001 to June 2003. We then asked participants to “find and remember everything *The New York Times* wrote about Robert Mugabe” using their tool. Because historians have only so much time for a given research project (Section 3.3.2), we limited participants to exactly six minutes to conduct their research using their assigned interface. After six minutes, we presented eight true/false reading comprehension questions about *New York Times* coverage of Mugabe and observed the total number of correct answers

¹⁹We did not ask participants to take the next step of drawing substantive historical conclusions from their findings, which would have required deep historical knowledge and specialized training.

for each participant. Because scoring well on this test of reading comprehension requires finding and reviewing information about a query in a corpus, we believe our crowd task measures how well people perform mention gathering and analysis, using a particular interface.

8.1.1 Details: Reading Comprehension Questions and Scoring. To ensure that our task was as objective and neutral as possible, we created reading questions using the Wikipedia page for Robert Mugabe [140]. Specifically, we used a semi-automated procedure based on tf-idf sentence vectors (described in detail in the Appendix) to identify Mugabe facts from Wikipedia reported in *New York Times* editorials about Mugabe. We then selected four facts from Wikipedia reported in the 12 editorials in the corpus and four facts from Wikipedia that were not reported in the 12 editorials. These four facts were reported in some other *New York Times* editorial that was not presented to participants (because the editorial was published before or after January 2001 to June 2003). In total, this process created a list of eight Mugabe facts from Wikipedia.

To evaluate reading comprehension, we presented all eight facts in randomized order and asked participants to select those facts that appeared in the articles they had reviewed during the task. To get a perfect score of eight out of eight correct answers without guessing,²⁰ a participant would have to find and remember the four Mugabe facts reported in the editorials shown during the task, without selecting any of the four facts that were not reported in the editorials. The Appendix includes a screenshot showing the reading comprehension questions.

8.2 Experiment Design and Experiment Details

We compared CLIOQUERY with a keyword document search system using a between-subjects experiment design with U.S. masters workers recruited via Amazon Mechanical Turk. (Amazon confers the master designation on crowdworkers with a record of success in crowd tasks.) Participants were randomly assigned to complete the reading comprehension task using either CLIOQUERY or a baseline keyword document search interface (IR). We then measured the difference in the mean number of total correct answers from workers in each group to determine if CLIOQUERY helped people find and remember information about Robert Mugabe (as compared to the IR system).

8.2.1 Implementation of the IR Baseline. We implemented the IR system using Whoosh, an open-source Python keyword document search tool that ranks results using the common BM25 metric.²¹ The Appendix contains a screenshot of this baseline interface.

To ensure fair comparison, we tuned Whoosh to be most similar to CLIOQUERY. Specifically, Whoosh accepts a number of configuration parameters that govern how the system creates snippets on the results page (see Section 2.2). Because such snippets are similar to the snippets in the CLIOQUERY Document Feed, we adjusted the Whoosh snippet parameters so Whoosh snippets were as close as possible in length to the shortened sentences in the CLIOQUERY Document Feed. Further details about the tuning procedure are described in the Appendix. We also adjusted the IR system to use the same font size as CLIOQUERY.

To minimize possible variation in worker behavior, we hard-coded the IR system (and the CLIOQUERY system) to use the query “Mugabe” during the experiment.²² To rank the 12 documents in the corpus using the IR system, we loaded the IR tool with all *New York Times* editorials published between 1987 and 2007 that include the word “Zimbabwe” and then queried for “Mugabe” while

²⁰In this task, a participant would have a $.5^8 * 100 = 0.391\%$ chance of correctly guessing all eight answers.

²¹Whoosh is similar to other traditional keyword document search tools like Lucene. <https://whoosh.readthedocs.io/>.

²²During the experiment, we also removed CLIOQUERY interface elements that are not relevant for the task, such as the corpus selection control, filter-by-date feature, and filter-by-subquery feature.

applying a date filter to select only those results that were published from January 2001 to June 2003.

We did not implement the IR system using proprietary black-box search tools such as Google or *New York Times* search [99]. This is because it is not possible to load open-source versions of such systems with a custom corpus. Loading a custom corpus is crucial for two reasons. (1) Our broader goal is to design an open-source software system that can be deployed and used by historians, who are often interested in corpora that are not published on the web (Section 10) and thus inaccessible to Google (or to any other web search engine product). Comparing to an open-source search engine is thus more appropriate than comparing to a black-box system like Google. A historian could use an open-source search engine to index and analyze the documents they collect during their work. (2) For a controlled experimental comparison of two user interfaces, it is necessary to fix the dataset used for both interfaces. In our experiment, users were evaluated based on what information they found (which would change based on the dataset).

8.2.2 Experiment Sequence, Experiment Pretest, and Phases of Data Collection. At the start of the experiment, participants in each condition watched a roughly one-minute training video describing how to use their randomly assigned interface. They also read several screens with task instructions, where they entered short phrases into text boxes to confirm they understood the task and were paying attention. After these preliminaries, participants took an easy pretest that was very similar to the main task (but was about Iraq instead of Zimbabwe). We describe the details of the pretest in the Appendix. After the pretest, participants proceeded to the main Robert Mugabe task, conducted their research, and answered the eight reading comprehension questions. The task concluded with qualitative questions, including questions about the strengths or weaknesses of the assigned interface. Qualitative questions are provided in the Appendix. In total, the task took 20 minutes.

Data collection for the task proceeded in two phases. We first collected data from 18 participants in a small initial pilot. Following the initial pilot, we made adjustments to the task described in the Appendix, including fixing a bug that was favorable to the IR baseline. Following these changes, we collected data from the remaining 103 participants. We decided to include data from the pilot in our analysis, because collecting data from crowdworkers was expensive and because we had trouble recruiting participants from the limited pool of master's workers. (Pooling data is common in settings where data is sparse.) Because we struggled with recruitment, we had to increase task payment from \$2.50 to \$5.00 during data collection. We include details in the Appendix.

8.2.3 Detecting Engaged and Not-engaged Workers. In their highly cited study on crowdsourcing for HCI, Kittur et al. [75] emphasize the importance of detecting suspect responses from crowdworkers who may not be completing tasks in good faith. We thus measure worker engagement in two different ways. First, because the pretest was designed to be very easy, we assume that participants who did not score perfectly on the pretest were less engaged in the crowd task than other participants. Second, we also assume that participants who made mistakes on task instructions were also less engaged. For instance, some participants made a mistake on task instructions by trying to skip ahead without watching the training video (we logged this and similar behaviors). In subsequent analysis, we refer to participants who both completed the pretest correctly and did not make any mistakes on task instructions as **engaged participants**. Engaged participants are a subset of **all participants**, the set of all people who completed the task.

8.3 Results and Analysis

We found that participants assigned to complete the historical reading comprehension task with CLIOQUERY averaged more total correct answers than participants assigned to complete the same

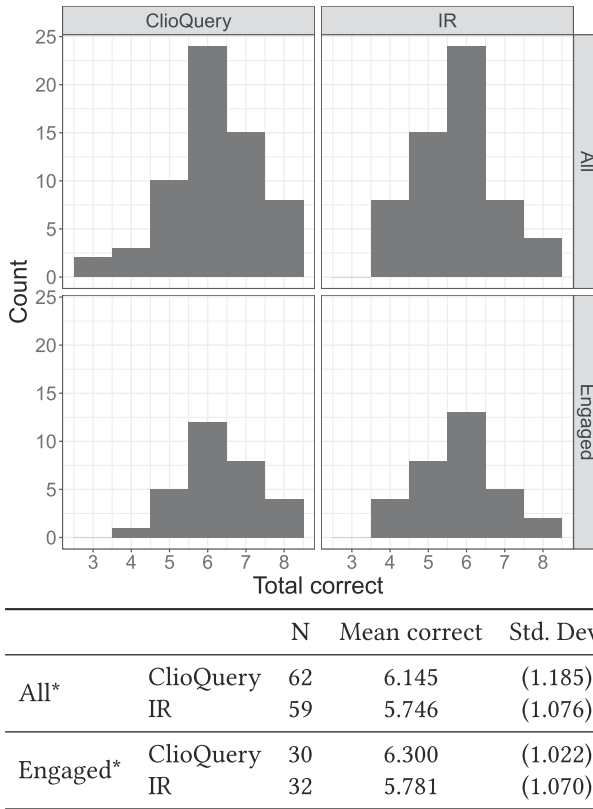


Fig. 7. Total correct questions by interface, among all participants, and among engaged participants. A star* indicates a significant difference between the means of the CLIOQUERY and IR groups.

task with the IR system (Figure 7). Among the 62 engaged participants, workers in the CLIOQUERY group averaged 0.519 more correct answers than workers in the IR group (Cohen's $d = 0.495$).²³ CLIOQUERY's effect was weaker among all participants, where CLIOQUERY workers averaged 0.399 more correct answers than IR workers (Cohen's $d = 0.352$). We hypothesize that this weaker effect may be due to inattention among non-engaged participants, which may have introduced data collection noise. For instance, participants who did not read task instructions carefully or who failed the pretest may have been more inclined to guess on the Mugabe task.²⁴

We tested for possible equality of means using bootstrap hypothesis testing [43] (Algorithm 16.2). Using 100,000 samples, we found that the difference in means among all workers in each condition was significantly different ($p = 0.030$). We also found that the difference in means among the subset of engaged workers in each condition was also significant ($p = 0.029$). We show separate bootstrapped distributions of sample means in Figure 8.

8.3.1 Qualitative Analysis. Our experiment suggested that some properties of the CLIOQUERY interface helped participants on the historical reading comprehension task. To try and gain a better

²³Computed with v0.8.1 of the effsize package in R.

²⁴The number of workers in each group is not exactly equal. This is common in crowdsourced settings, where some workers may not finish a task. For instance, we used an alert to ask workers attempting to complete our task on a phone rather than a computer to not proceed with the survey.

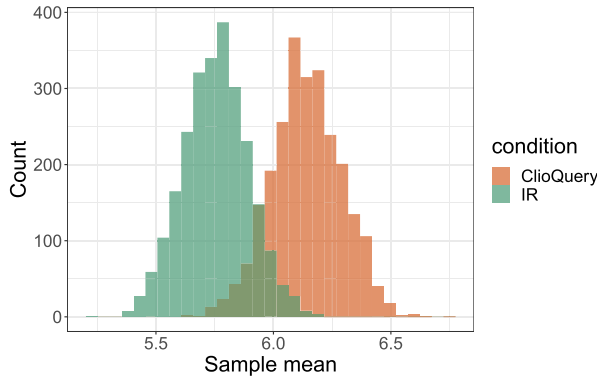


Fig. 8. Distribution of sample means, across 2,500 bootstrap samples of scores from the 59 IR participants and 2,500 separate bootstrap samples of scores from the 62 CLIOQUERY participants.

understanding of which exact aspects of CLIOQUERY may have been helpful, we reviewed qualitative feedback from the eight CLIOQUERY participants who achieved a perfect score in reading comprehension. Each of these participants praised one or more CLIOQUERY features in offering qualitative feedback on the system. *“I liked that I could expand the articles and filter them by the number of times the key word was mentioned,”* one top performer wrote. Said another, *“I liked being able to control the number of mentions so I could determine relevance rather than trust a search engine.”* A third liked that CLIOQUERY, *“made it easy to see which articles I have already read and which ones I have yet to read.”* One high scorer did note that while in-text highlighting was in general helpful, *“the part of this highlighting that I didn’t like is that ... it was hard to gain context without reading the unhighlighted text before or after the highlighted sections.”*

However, the four IR users who got perfect scores offered scattered feedback. One liked how the results did not show *“a bunch of random stuff or products to buy,”* while two others disagreed if the snippets were useful (one praised them, one said they did not help). Comments from the final high-scoring IR participant suggested that the IR system offered a realistic baseline. *“There wasn’t much to like or dislike,”* they said. *“I really didn’t find any differences how I would normally do it.”*

9 DISCUSSION

9.1 CLIOQUERY Suggests New Features and Directions for Interactive Text Analysis

Much prior work in interactive text analysis focuses on helping people investigate bodies of documents by presenting textual units such as topics [78], events [82], or thematic hierarchies [13]. But motivated by the needs of historians and archivists, CLIOQUERY instead proposes and tests a new approach to interactive corpus investigation, organized around the analysis of query mentions in context (see Section 2.3).

To help people investigate this “unit of analysis” [24], CLIOQUERY employs new text summarization techniques from the NLP literature to create a summary of a query term across a corpus. The system then presents summaries alongside more traditional features from the text analytics literature, such as linked views and in-text highlighting, to help people easily and transparently review summary text in underlying documents. During expert interview and field-study evaluations, many historians said that they found such features helpful for archival research. They reported skimming over query mentions in the Document Feed to gain a sense of a query’s use across a corpus and then reading highlighted mentions in the Document Viewer for more context and detail. Several specifically mentioned that these components helped with mention gathering and analysis.

This query-oriented approach suggests new directions for interactive text analytics in other query-oriented settings. For instance, some marketing applications identify salient words and phrases in online forums [52, Section 4.1]; CLIOQUERY’s query-focused summaries and linked views might help marketing analysts using such systems understand what people say about products online. Features based on CLIOQUERY might also be applied in existing text analytics systems. For instance, people might formulate a query using overview-oriented features such as word clusters and then investigate this query word using a CLIOQUERY-style Document Feed and Document Viewer.

9.2 CLIOQUERY Tests an Idea: “Text and Its Affordances Should Be Taken Seriously”

Researchers have proposed many approaches to text visualization, which map high-dimensional text to two-dimensional graphical representations such as time series plots (e.g., ThemeRiver [61]) or bubble diagrams (e.g., EventRiver [82]). By contrast, CLIOQUERY’s Document Feed and Document Viewer do not map text data to a graphical representation. Instead, CLIOQUERY uses text summarization methods from NLP to extract and present spans of text from a corpus for people to read, using automatic in-text highlighting to facilitate skimming. In this sense, CLIOQUERY follows the advice of Sultanum et al. [130], who suggest that “text and its affordances should be taken seriously” in text analytics by making text itself “a central piece of the visualization.” Viewed through the lens of this recommendation, CLIOQUERY reflects one strategy for a text analytics system fundamentally organized around displaying spans from a corpus. Other work from Sultanum et al. [129] also explores this “reading-centered approach.” Some authors of prior text analytics systems have later noted the importance of showing underlying documents during interactive analysis. Authors of the Jigsaw system found that “interactive visualization cannot replace the reading of reports” [53]. Similarly, creators of both Overview [127, Section 5] and ThemeRiver [61, Section 7] also describe finding that people need to read underlying text.

9.3 User Feedback on Summarization Has Implications for Natural Language Processing

CLIOQUERY applies particular ideas from query-focused text summarization for interactive text analysis. However, building and evaluating a user-facing system forced us to reexamine several core assumptions from the text summarization literature. In particular, early versions of CLIOQUERY applied standard optimization-based summarization methods [90] to select “important” information from a corpus. This approach was reminiscent of prior temporally oriented language engineering systems such as HistDiv [124], TimeMine [2], and TimeExplorer [88], which each attempt to automatically identify most-relevant information based on a query.

However, during needfinding and prototyping, we found that some historians and archivists strongly disliked this approach. Experts reported that they needed to understand why the computer was showing particular summaries before they could actually draw conclusions from the output (see prototypes in the Appendix). Based on this feedback, in later versions of CLIOQUERY, we stopped trying to extract “important” mentions of a query term in search results. Instead, we decided to shorten and present every single sentence mentioning a user’s query in the Document Feed and allow people to easily examine such shortenings in context in the Document Viewer. During our expert interview and field study and evaluations, we found that this approach was more successful. We hypothesize that experts liked this format because they could understand why CLIOQUERY showed query shortenings and thus use CLIOQUERY output in their research.

Our experiences might have implications for NLP, where research in summarization typically focuses on generating summaries that best match “gold” references [35, 97] without worrying

about explaining how summaries are formed. In particular, much recent work on abstractive summarization in NLP [65, 113] seeks to generate summary passages that do not occur in the input text. Because such abstractive output can not be checked against underlying sources, and because such methods also currently suffer from frequent factual errors [77], much more research may be required before abstractive approaches might be applied towards social research.

9.4 Comprehensive and Unbiased Search Costs Time; Transparency Might Help

During needfinding interviews, historians and archivists often emphasized the importance of directly and comprehensively examining all evidence relevant to a given research question, without allowing black-box algorithms to influence their conclusions. We thus designed CLIOQUERY to minimize potential bias from algorithmic ranking. Yet, feedback on these aspects of CLIOQUERY was mixed (Sections 6.3 and 6.5). Some appreciated how CLIOQUERY used filters instead of ranking to narrow down search results. But others reported that truly forgoing algorithmic curation required the researcher to spend too much time reading irrelevant documents. For instance, some admitted that they often have no choice but to trust computer models of relevance to find evidence in archives, because keyword search often turns up far more documents than they can possibly review. While historians do sometimes work with smaller corpora (Section 4.3), this issue would be particularly problematic in larger archives, where some queries will be mentioned many times.

Why did some express deep commitment to full manual review of evidence during needfinding interviews, while others admit that they had to trust search engines to select evidence during system evaluation? There are at least two possibilities. One possibility is that historians and archivists might express commitment to comprehensive review when describing their ideal practices but remember the limitations of this ideal when faced with a real task during system evaluation. Some approaches to needfinding in HCI emphasize the limits of user interviews [11], because “what people say and what they do can vary significantly.” Another possibility is that there is variation in historians’ commitment to comprehensiveness. Some but not all historians may feel required to comprehensively review all evidence during research, possibly based on intellectual background or subfield. (Other authors find similar variation among doctors [131, Section 4.3.5].) Better understanding this apparent contradiction between experts’ stated commitments to comprehensive review and the realities of inevitable tradeoffs between recall and time [108, Figure 6] will require further research.

Nevertheless, future researchers might resolve the contradiction with improved user interfaces. Specifically, systems might transparently show which documents are selected or hidden by an algorithm and allow people to easily override and investigate any document-ranking decisions from a machine. Such features would be particularly important for larger corpora, where historians would not be able to review all query mentions in context. Research on tools for visually and interactively refining search results [118] might offer a useful starting point. Features that help groups of historians to collaborate during search could also enable teams of researchers to comprehensively review evidence from larger corpora.

10 LIMITATIONS AND FUTURE WORK

Because it was difficult and expensive to recruit and interview highly trained experts, this study relies on in-depth interviews with a small sample of humanists. While such one-on-one interviews provided rich feedback, the opinions of our participants likely only approximate the true requirements of all historians and archivists. Moreover, interview studies may have limitations in unearthing design requirements (Section 9.4). In the future, we thus plan to take steps to facilitate adoption to learn more about user needs. In particular, we found that historians have to

collect, organize, and sometimes digitize news stories before they are ready to gather and analyze query mentions (Section 6.6). We thus plan to add features for importing news stories into CLIOQUERY from existing tools such as Zotero and the Internet Archive. Additionally, throughout this work, we assume that query mentions are defined by exact string matches. This simplifying assumption allows us to focus on user experience and interaction but has clear limitations. For instance, authors sometimes refer to “Reagan” using the nickname “Dutch.” Automatically detecting such aliases (and other deviations from exact string matching) will be important for future work.

11 CONCLUSION

This study describes the design and evaluation of the CLIOQUERY text analytics system. Where prior tools focus on the analysis of textual units such as topics [78], events [82], or hierarchies [13] (Section 2.3), CLIOQUERY is uniquely organized around investigation of query words in context, which form the system’s central “unit of analysis” [24].

CLIOQUERY’s unusual emphasis on the analysis of query words in context emerged from our study into the needs and practices of historians and archivists who find and analyze occurrences of queries in their research. Working with and studying historians revealed that analyzing change across time, undertaking comprehensive review of evidence, evaluating contextual information, and conducting neutral observation were each central to the practice of historical research. Based on these insights, we designed the CLIOQUERY system, which applied query-focused text summarization methods from NLP to create skimmable summaries of a query term across an archive. CLIOQUERY then used more traditional analytics features like linked views and automatic in-text highlighting to show summary text within the context of underlying news stories to build expert trust in automatic summaries.

We tested CLIOQUERY in two separate user studies with historians, where we found that CLIOQUERY’s approach to organizing and presenting query mentions could help experts answer real questions from news archives. Many historians reported that CLIOQUERY’s Document Feed facilitated rapid analysis of query mentions and that CLIOQUERY’s linked Document Viewer offered complementary context and detail. In a separate quantitative comparison study, we found that CLIOQUERY helped crowd participants answer significantly more questions than a keyword document search tool.

Together, our work on CLIOQUERY suggests possible new directions for interactive text analysis. In particular, CLIOQUERY’s combination of text summarization and linked in-text highlighting could be applied in other query-oriented settings, where people also need to investigate query words in context. For instance, some marketing applications suggest notable keywords from comments in online forums [52, Section 4.1]. CLIOQUERY methods might be applied to help marketers gather and analyze keyword mentions or to help others investigate queries in other domains.

A APPENDIX

The CLIOQUERY system: Additional Details

Implementation Details. CLIOQUERY is a web application written in Python 3, using the Flask and React libraries.²⁵ The text simplification methods in the article use Stanford CoreNLP [85] for tokenization, dependency parsing, and part-of-speech tagging.²⁶ CLIOQUERY’s relationship span

²⁵<https://flask.palletsprojects.com/en/1.1.x/> and <https://reactjs.org/>.

²⁶Eisenstein [45] offers a detailed introduction to these NLP techniques.

extraction method also employs logistic regression; we use the implementation from Scikit-learn [105]. In the future, rewriting our Python-based prototype in a faster language like Java or C would reduce our system's latency, helping CLIOQUERY scale to larger corpora. It might also be possible to further improve performance by employing time and space efficient IR methods for efficiently indexing and retrieving the locations of query words in documents [84].

Time Series View: Additional Details. CLIOQUERY's Time Series View shows a single rug point (small vertical line) for each document mentioning the query. These markings both help explain aggregated count statistics encoded in the time series plot (more rug points mean an higher annual count) and help link the Time Series View with the Document Feed. If a user hovers over a rug point, then CLIOQUERY displays the headline of the corresponding news story using a tooltip; if the user clicks a rug point, then CLIOQUERY updates so the story is displayed in the Document Feed and in the Document Viewer. When a user hovers over some year in the Time Series View, CLIOQUERY displays a tooltip showing the total count of documents containing the query for that year.

Default System Behaviors. If a user has not yet entered a query, then CLIOQUERY's time series plot simply shows the overall counts of documents by year across the entire corpus, shown with a neutral black line. In this case, the Document Feed also shows all documents in the corpus. Moreover, when filter-by-date is not used, CLIOQUERY shows documents from the time span of the corpus.

Choosing Colors. We chose colors for CLIOQUERY using Colorbrewer [14], a common resource, which offers colorblind-safe and print-friendly palettes. Hall and Hanna [56] test how foreground and background color affects how people read, retain, and experience text on screen. Our study focuses on testing the utility of in-text highlighting and text simplification for expert social researchers; future work might test the effect of varying the foreground or background color.

Handling Token Gaps during Clause Deletion. In some cases, there may be gaps between tokens in simplified mentions, where tokens have been removed from the middle of a sentence (these are shown with ellipses in the Document Feed). In these cases, in performing automatic in-text highlighting to link the Document Feed and Document Viewer, we highlight the span in the Document Viewer, which begins with and ends with the first and last token of the corresponding simplified mention, shown in the Document Feed.

Computing Tf-idf Scores of Iterative Clause Deletion. To compute tf-idf scores during iterative clause deletion, we assign each word in each possible output candidate shortening a word-level tf-idf score and average the word-level tf-idf scores of all words in each possible candidate shortening to compute an overall, sentence-level tf-idf score. We assign each word a tf score equal to the total occurrences of the word among all documents that contain Q and an idf score equal to 1 divided by the count of documents containing the word across the corpus. We then multiply each word's tf score by its idf score to get a word-level td-idf score. We then select the candidate shortening with the highest overall tf-idf score for display in the Document Feed.

Choosing among Possible Sentence-shortening Methods. In the System section, we describe three different sentence-shortening techniques, which are applied in the CLIOQUERY interface. Below, we describe how CLIOQUERY chooses to apply the three different methods.

After a user enters a query Q , for each document mentioning Q , CLIOQUERY's Document Feed displays the first sentence within the document mentioning Q that can be shortened via query-focused clause deletion. If no such sentence exists, then CLIOQUERY resorts to shortening the first

sentence mentioning Q via character windowing. (Character windowing is only used as a last resort, because it does not attempt to create well-formed output containing salient words from the input.)

In cases when a user has entered both a query and subquery, for each document mentioning the query or subquery, CLIOQUERY will attempt to display the first sentence in the document that can be shorted via relationship span extraction. This is because we assume the user is interested in the relationship between the query and subquery. If there is no sentence that can be shorted via relationship span extraction, then CLIOQUERY will display the first sentence that can be shortened via query-focused clause deletion. If no sentence can be shortened via clause deletion, then it will resort to shortening the first sentence mentioning the query or subquery via character windowing.

CLIOQUERY also allows the user to click “expand” to see all sentences mentioning the query within the document, as described in the System section. In this case, CLIOQUERY will first attempt to shorten each sentence mentioning Q via query-focused clause deletion before resorting to shortening the sentence with character windowing. If the user has also set a subquery (in addition to Q), then CLIOQUERY will first try to shorten each sentence mentioning the query and subquery using relationship span extraction (and then attempt clause deletion and character windowing).

Field Study: Additional Details

To help $H1$ answer their question using CLIOQUERY, we gathered a custom corpus of articles from the *New York Times* (NYT). To gather the corpus, we searched for “El Salvador” on the *New York Times* website [99] and then automatically downloaded all query-matching articles published between 1980 and 1985 in the “World News” and “Week in Review” sections of the newspaper. We filtered downloaded articles to create a corpus of NYT articles containing the word “Salvador,” and we loaded this corpus into CLIOQUERY for $H1$.

To help $H2$ answer their research question, we similarly gathered a second custom corpus of articles by searching for “astronaut” on the *New York Times* website [99] and then automatically downloading all query-matching articles published between 1980 and 1985. We then similarly filtered the documents to ensure that all query-matching mentioned “astronaut” and loaded the corpus into CLIOQUERY for $H2$.

Quantitative Comparison Study: Additional Details

Additional Details Regarding Creation of Reading Comprehension Questions. We used a semi-automated procedure to create reading comprehension questions for our quantitative crowd study. Specifically, we first collected all editorials from the New York Times Annotated Corpus [114], which included the words “Zimbabwe” and “Mugabe.” We then used the TfidfVectorizer class from scikit-learn [105] with default settings to construct tf-idf vectors for all 1,689 sentences in the editorials. We also similarly constructed tf-idf vectors for all 597 sentences from the Wikipedia page on Robert Mugabe [140]. We then computed the cosine similarity of each sentence pair in the Cartesian product of Wikipedia and *New York Times* sentences. We manually reviewed the 200 sentence pairs with the highest cosine similarities and manually labeled 37 total sentences from *New York Times* editorials that reported a fact described in some sentence from Wikipedia. This process identified 37 facts about Mugabe from Wikipedia reported in editorials in *The New York Times*. We selected 8 of these facts to create reading comprehension questions for our task.

Additional Details Regarding Tuning of IR Baseline. We implemented the IR baseline using Whoosh, an open-source Python search engine. Like many search engines, Whoosh shows small

document snippets from ranked documents on the search engine results page (Figure A.3). To encourage fair comparison between Whoosh and CLIOQUERY, we tuned Whoosh so document snippets contained roughly as much text as the shortened sentences in the CLIOQUERY Document Feed. Specifically, Whoosh allows snippet customization by setting the `maxchars` and `surround` parameters in its Highlighter module. We set these parameters by performing a grid search over all possible values from 10 to 100 (for each parameter) to maximize the average number of characters per Whoosh document snippet, under the constraint that the average was less than or equal to 90 characters (the length of the longest-possible shortened sentence in the CLIOQUERY Document Feed). The final setting for the `surround` parameter was 27 characters and the final setting for the `maxchars` parameter was of 10 characters. Using these settings, we observe a mean snippet length of exactly 90 characters using the IR system on the crowd task. Beyond tuning these parameters, we use default settings for the Whoosh search engine.

Additional Details Regarding the Crowd Study Pretest. Before beginning the main task in our crowd study, participants in each condition used their interface to complete a three-minute pretest using a small corpus of six *New York Times* editorials mentioning “Iraq.” The pretest was very similar to the main task; each interface was hard-coded to use the query “Falluja,” and participants were instructed to “find and remember everything the *New York Times* wrote about Falluja” using their tool. After participants typed this exact phrase into a text box to confirm they understood the instructions, they conducted research using their assigned interface. After three minutes, participants were then presented with a screen with four facts about U.S. involvement in Falluja (included in supplemental materials) and asked to identify which facts were reported in the six articles. Because only one fact from the list was reported in the articles, to get a perfect score of 4 out of 4 on the pretest, workers had to both correctly identify the reported fact, and refrain from guessing any of the other three facts. The pretest was designed to be very easy for attentive workers.

Additional Details Regarding Data Collection Phases for the Crowd Task. Data collection for the crowd task proceeded in two phases: an initial pilot phase and a main data collection phase. After the small pilot, we added two training screens for CLIOQUERY participants (shown in supplemental materials) to help CLIOQUERY users gain practice using unfamiliar features. We also fixed a bug in the pilot in which CLIOQUERY users were shown an extra two editorials. We emphasize that these two editorials did not contain any facts about Mugabe that could be used to answer the reading comprehension questions, and also note that the two extra editorials would have made the task harder for CLIOQUERY participants (because they would have had to read extra text during the task, which was not relevant to the reading comprehension questions). Finally, after the pilot, we adjusted the random assignment mechanism so participants were assigned to conditions in an alternating fashion following an initial random draw (i.e., first CLIOQUERY, then IR, then CLIOQUERY...). In the pilot, participants were assigned to conditions at random when they loaded the first screen in the task.

Additional Details Regarding Task Payment. Because we had trouble recruiting qualified masters workers for our lengthy and complex task, we increased payment during data collection. The first 18 participants were paid \$2.50 to complete the pilot. After the pilot, we increased payment to \$3.00 and collected data from 75 more participants. Because data collection was still very slow (e.g., 10 workers over a 24-hour period), we further increased payment to \$4.00 for the task and collected data from 26 more workers. Finally, we increased payment to \$5.00 for the task. When only 2 workers signed up over a half-day period at the \$5.00 rate, we ended data collection.

A.1 Additional Figures

You have used the tool for 6 minutes and have been automatically directed to this page for the next phase of the study.

Based on your research, what did the New York Times write about Mugabe? Check **all** that apply.

- ☐ South African Archbishop Desmond Tutu described Mugabe as a dictator
- ☒ The European Union imposed sanctions on Mr. Mugabe
- ☐ Mugabe called on supreme court justices to resign
- ☒ Mugabe called Zimbabwe's white farmers "terrorists"
- ☒ Leaders of Malawi and Mozambique have spoken out against Mr. Mugabe
- ☒ Mugabe blamed Britain for problems in Zimbabwe
- ☐ Mugabe sent troops to prop up dictator Laurent Kabila
- ☐ Mugabe signed an agreement with Joshua Nkomo

Fig. A.1. Participants answered eight true/false questions about what the *New York Times* wrote about Robert Mugabe, using the form shown above. The four facts shown with checkboxes were described in editorials available to participants during the study. The four false facts shown without checkboxes were described in other editorials, not available to participants during the study. Participants who found and remembered the four facts from the corpus and who also did not incorrectly guess any of the four facts not described in the corpus scored 8 out of 8 on the reading comprehension task. The order of questions was randomized.

Final questions
 Thanks for your participation. Please answer a few final free response questions.

Please provide a 2 or 3 sentence summary of how the New York Times portrayed Robert Mugabe in these editorials. Include the most important details that can remember in your response.
 e.g. "Robert Mugabe prevented opposition leader Morgan Tsvangirai from taking control of Zimbabwe. This drew condemnation from western leaders like Colin Powell."

Think of a time you used a computer to find information from documents. Describe what information you were looking for and how you found it.
 e.g. "I looked through reviews of restaurants online" or "I looked up information about Ireland using library databases for a school assignment"

Compare those past experiences finding information with your experiences in this task. Was there anything you liked or disliked about the system you just tried?
 e.g. "I liked how it showed text fragments to help me understand which information was in the document" or "I don't think the tool would help for restaurant reviews."

Do you have any feedback for us about this Amazon Turk Task?
 e.g. "I needed more time to read all the articles" or "task was easy"

Fig. A.2. Qualitative questions for participants at the end of the crowd task.

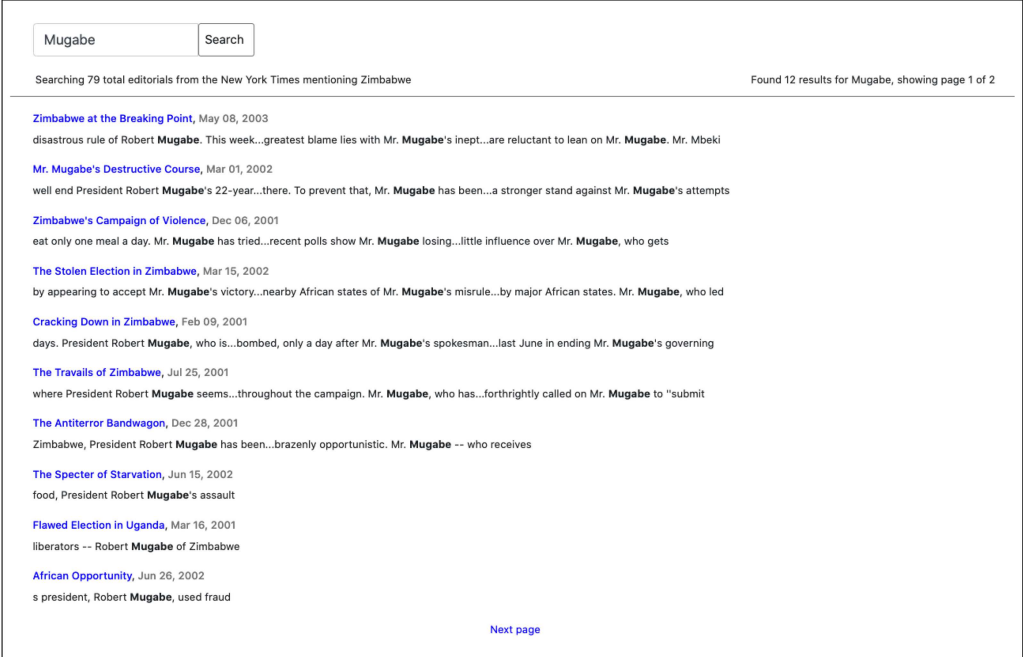


Fig. A.3. The IR baseline interface in our crowd study.

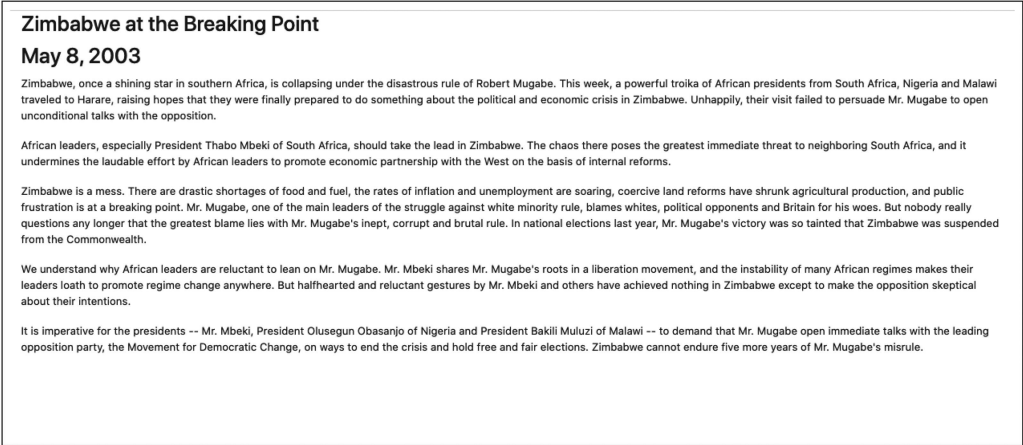


Fig. A.4. A single search result from the IR interface in our crowd study.

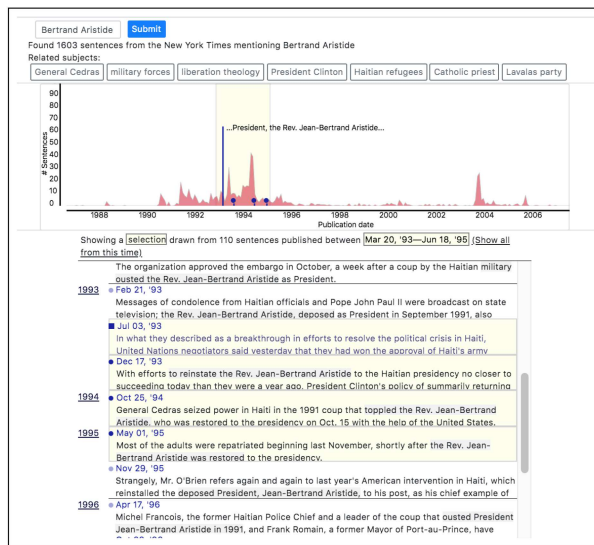


Fig. A.5. An early prototype of CLIOQUERY, which used traditional, optimization-based text summarization methods from natural language processing [90] to try and select the most salient information from a given time period for display. This prototype selects four most “important” articles (shaded in light yellow in the feed below) to summarize the hundreds of articles mentioning the query “Bertrand Aristide” in the *New York Times* from May 20, '93 to June 18, '95 (shown shaded in light yellow in the time series above). I5 strongly disliked this approach, prompting a shift towards interfaces emphasizing transparency and trustworthiness. “I need to know what is included and why,” I5 explained. “I need to know why it is showing this limited view.” I5 continued, “I am wary of algorithms that choose for me what the important facts are. I am a PhD historian. Leaving stuff out. We are taught to be critical of that.” Ultimately, I5 noted, “History is written by the victors. What actually matters is what people choose to put in the timeline.” We theorize that I5 could not trust the prototype because it seemed to lack the capacity to select important facts or the integrity to adhere to historical research principles; prior work in HCI (e.g., SMILY [18]) assumes that to earn user trust, a system must have both the capacity to help the user and the integrity to adhere to principles that are important in a given domain.

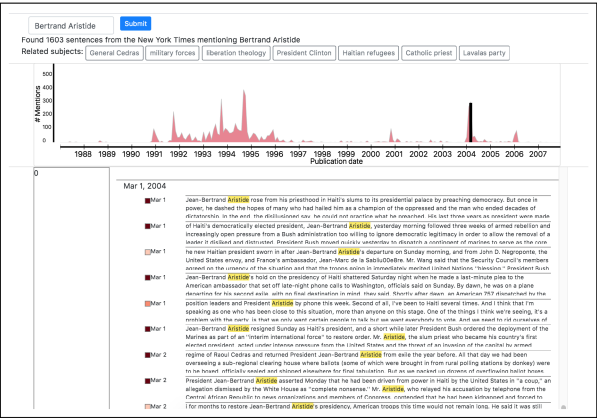


Fig. A.6. An early prototype of CLIOQUERY, displaying and highlighting every single mention of the query term “Aristide” in *New York Times* articles mentioning “Haiti.” “Are we showing too much information in this interface?” one researcher from our group asked I4, when presenting the prototype. “This is literally every mention of your query term.” “No this is good,” I4 explained, “because of what I was calling the type II error concern [i.e., the fear of missing relevant material]. When I see something that is trying to decide or curate for me that is a worry. That is a red flag.” However, I4 went on to explain how the interface needed to provide more context and transparency surrounding highlighted snippets. “With this design you have to click or read each snippet to see if it is relevant,” he said. “The snippets are valuable and good but very small and you have to look at the contents of the article. Sometimes you can eliminate that by just quickly scanning the article title ... there needs to be a way to provide the information in a more transparent way.” (The search bar shown at the top is non-functioning mockup; the “0” on the left-hand side is a placeholder.)

A.2 Additional Tables

Table A.1. Interview Study Participants

ID	Research experience	Library experience	University role	Gender	Topic
P1	6	0	PhD candidate	Male	Iraq
P2	5	0	PhD candidate	Male	Zimbabwe
P3	4	0	PhD candidate	Female	combat
P4	20	0	Instructor/researcher	Female	wages
P5	0	3	History librarian	Male	copyright

We report history and library experience in years.

Table A.2. Historians in the Field Study

ID	Research experience	Library experience	Academic role	Gender	Research area
H1	5	0	PhD student	Male	Media and society
H2	25	0	Tenured faculty	Female	Space exploration

History and library experience are listed in years.

Table A.3. Interviewees in our Needfinding Study

ID	Research experience	Library experience	University role	Gender	Field
I1	5–10	1–5	PhD Candidate	Male	History
I2	0	20–30	Librarian	Female	Lib. Science
I3	10–20	0	Junior Faculty	Female	Am. Studies
I4	10–20	10–20	Archivist	Male	Lib. Science
I5	10–20	10–20	Librarian	Non-binary	History

We list research experience and library experience as a range of years. We abbreviate American Studies as Am. Studies and Library Science as Lib. Science.

Table A.4. A Selection from Prior Work in Library Science and Information Science, Focused on the Information-seeking Behavior of Historians

Author(s)	Venue	Study type	Participants
Allen and Sieczkiewicz [2]	<i>Proc. ASIS&T</i> (Info. Science)	Interview	8
Case [19]	<i>The Library Quarterly</i>	Interview	20
Duff and Johnson [40]	<i>The Library Quarterly</i>	Interview	10
Chassanoff [20]	<i>The American Archivist</i>	Survey	86
Dalton and Charnigo [34]	<i>College & Research Libraries</i>	Survey	278
Duff, Craig, and Cherry [39]	<i>The Public Historian</i>	Survey	600

These papers describe studies of $N = 1,002$ historians (in total).

ACKNOWLEDGMENTS

We thank Mahmood Jasim for detailed feedback throughout our research process. We also thank Su Lin Blodgett, Javier Burroni, Katherine A. Keith, and Kalpesh Krishna for helpful discussions, suggestions, and edits. Finally, we thank Alyx Burns, Lucy Cousins, Prachi Modi, and Ali Sarvghad for offering feedback on the user interface.

REFERENCES

[1] Eric Alexander, Joe Kohlmann, Robin Valenza, Michael Witmore, and Michael Gleicher. 2014. Serendip: Topic model-driven visual exploration of text corpora. In *VAST*.

[2] Robert B. Allen and Robert Sieczkiewicz. 2010. How historians use historical newspapers. In *Proc. ASIS & T. American Society for Information Science*.

[3] Joshua Bambrick, Minjie Xu, Andy Almonte, Igor Malioutov, Guim Perarnau, Vittorio Selo, and Iat Chong Chan. 2020. NSTM: Real-time query-driven news overview composition at Bloomberg. In *ACL: System Demonstrations*.

[4] Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computat. Ling.* 31, 3 (2005).

[5] Eric P. S. Baumer, David Mimno, Shion Guha, Emily Quan, and Geri K. Gay. 2017. Comparing grounded theory and topic modeling: Extreme divergence or unlikely convergence? *J. Assoc. Inf. Sci. Technol.* 68, 6 (2017), 1397–1410.

[6] Sherry M. Bell and R. S. McCallum. 2016. *Handbook of Reading Assessment: A One-stop Resource for Prospective and Practicing Educators* (2nd ed.). Routledge, New York.

[7] Nicolas Garcia Belmonte. 2014. Extracting and visualizing insights from real-time conversations around public presentations. In *IEEE VAST*.

[8] Emily M. Bender. 2013. Linguistic fundamentals for natural language processing: 100 essentials from morphology and syntax. *Synth. Lect. Hum. Lang. Technol.* 6, 3 (2013).

[9] William R. Black. 2018. How watermelons became black: Emancipation and the origins of a racist trope. *J. Civ. War Era* 8, 1 (2018), 64–86.

[10] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *J. Mach. Learn. Res.* (2003).

- [11] Jeanette Blomberg and Mark Burrell. 2012. An ethnographic approach to design. In *The Human-computer Interaction Handbook*, Julie A. Jacko (Ed.). CRC Press, New York.
- [12] Lauren Bradel, Chris North, Leanna House, and Scotland Leman. 2014. Multi-model semantic interaction for text analytics. In *VAST*.
- [13] M. Brehmer, S. Ingram, J. Stray, and T. Munzner. 2014. Overview: The design, adoption, and analysis of a visual document mining tool for investigative journalists. *IEEE Trans. Visualiz. Comput. Graph.* 20, 12 (2014).
- [14] Cynthia Brewer. 200x. ColorBrewer. (200x).
- [15] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. In *WWW*.
- [16] Erika Bsumek, Matthew O'Hair, Ian Diaz, and Braeden Kennedy. 2020. ClioVis. Retrieved from <https://web.archive.org/web/20201224213752/>.
- [17] Andreas Buja, John Alan McDonald, John Michalak, and Werner Stuetzle. 1991. Interactive data visualization using focusing and linking. In *Proceedings of the 2nd Conference on Visualization*. IEEE.
- [18] Carrie J. Cai, Emily Reif, Narayan Hegde, Jason Hipp, Been Kim, Daniel Smilkov, Martin Wattenberg, Fernanda Viegas, Greg S. Corrado, Martin C. Stumpe, and Michael Terry. 2019. Human-centered tools for coping with imperfect algorithms during medical decision-making. In *CHI*.
- [19] Donald Owen Case. 1991. The collection and use of information by some American historians: A study of motives and methods. *Libr. Quart.* 61, 1 (1991), 61–82.
- [20] Alexandra Chassanoff. 2013. Historians and the use of primary source materials in the digital age. *Amer. Archiv.* 76, 2 (2013), 458–480.
- [21] Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*.
- [22] Daniel Chomsky and Scott Barclay. 2010. The mass media, public opinion, and lesbian and gay rights. *Ann. Rev. Law Soc. Sci.* 6, 1 (2010), 387–403.
- [23] Jason Chuang, Christopher D. Manning, and Jeffrey Heer. 2012. Termite: Visualization techniques for assessing textual topic models. In *AVI*.
- [24] Jason Chuang, Daniel Ramage, Christopher Manning, and Jeffrey Heer. 2012. Interpretation and trust: Designing model-driven visualizations for text analysis. In *CHI*.
- [25] Charles L. A. Clarke, Eugene Agichtein, Susan Dumais, and Ryen W. White. 2007. The influence of caption features on clickthrough patterns in web search. In *SIGIR*.
- [26] James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *J. Arti. Intell. Res.* 31 (2008), 399–429.
- [27] Chrome Contributors. 2021. Search the web on Chrome. Retrieved from <https://web.archive.org/web/20201105231921/>.
- [28] Internet Archive Contributors. 200x. Internet Archive. (200x).
- [29] Lucene contributors. 2020. Apache LuceneTM 7.4.0 Documentation. Retrieved from <https://lucene.apache.org/core/740/index.html>.
- [30] Omeka Contributors. 200x. Omeka. (200x).
- [31] W. Bruce Croft, Donald Metzler, and Trevor Strohman. 2015. *Search Engines: Information Retrieval in Practice*. Pearson Education.
- [32] Weiwei Cui, Shixia Liu, Li Tan, Conglei Shi, Yangqiu Song, Zekai Gao, Huamin Qu, and Xin Tong. 2011. TextFlow: Towards better understanding of evolving topics in text. *IEEE Trans. Visualiz. Comput. Graph.* 17, 12 (2011).
- [33] Weiwei Cui, Shixia Liu, Zhuofeng Wu, and Hao Wei. 2014. How hierarchical topics evolve in large text corpora. *IEEE Trans. Visualiz. Comput. Graph.* 20, 12 (2014).
- [34] Margaret Stieg Dalton and Laurie Charnigo. 2004. Historians and their information sources. *Coll. Res. Libr.* 65, 5 (2004).
- [35] Dipanjan Das and André F. T. Martins. 2007. *A Survey on Automatic Text Summarization*. Technical Report. Carnegie Mellon University.
- [36] N. Diakopoulos, M. Naaman, and F. Kivran-Swaine. 2010. Diamonds in the rough: Social media visual analytics for journalistic inquiry. In *IEEE VAST*.
- [37] Anthony Don, Elena Zheleva, Machon Gregory, Sureyya Tarkan, Loretta Auvil, Tanya Clement, Ben Shneiderman, and Catherine Plaisant. 2007. Discovering interesting usage patterns in text collections: Integrating text mining with visualization. In *CIKM*. 10.
- [38] Wenwen Dou, Li Yu, Xiaoyu Wang, Zhiqiang Ma, and William Ribarsky. 2013. HierarchicalTopics: Visually exploring large text collections using topic hierarchies. *Trans. Visualiz. Comput. Graph.* 19, 12 (2013).
- [39] Wendy Duff, Barbara Craig, and Joan Cherry. 2004. Historians' use of archival sources: Promises and pitfalls of the digital age. *Pub. Histor.* 26, 2 (2004), 7–22.

- [40] Wendy M. Duff and Catherine A. Johnson. 2002. Accidentally found on purpose: Information-seeking behavior of historians in archives. *Libr. Quart.* 72, 4 (2002), 472–496.
- [41] M. Dörk, S. Carpendale, C. Collins, and C. Williamson. 2008. VisGets: Coordinated visualizations for web-based information exploration and discovery. *IEEE Trans. Visualiz. Comput. Graph.* 14, 6 (2008).
- [42] Marian Dörk, Nathalie Henry Riche, Gonzalo Ramos, and Susan Dumais. 2012. PivotPaths: Strolling through faceted information spaces. *IEEE Trans. Visualiz. Comput. Graph.* 18, 12 (2012).
- [43] Bradley Efron and Robert Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman & Hall.
- [44] Maud Ehrmann, Estelle Bunout, and Marten Düring. 2019. Historical newspaper user interfaces: A review. In *International Federation of Library Associations World Library Information Conference*.
- [45] Jacob Eisenstein. 2019. *Introduction to Natural Language Processing*. The MIT Press, Cambridge, MA.
- [46] Henry Feild, Ryan W. White, and Xin Fu. 2013. Supporting orientation during search result examination. In *CHI*.
- [47] Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with LSTMs. In *EMNLP*.
- [48] Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *EMNLP*.
- [49] Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *INLG*.
- [50] Eric Foner. 1995. *Free Soil, Free Labor, Free Men: The Ideology of the Republican Party before the Civil War*. Oxford University Press, New York.
- [51] Lisa Given and Kristie Saumure. 2008. Convenience sample. In *The SAGE Encyclopedia of Qualitative Research Methods*, Lisa Given (Ed.). SAGE Publications, Inc., Thousand Oaks, CA.
- [52] Natalie Glance, Matthew Hurst, Kamal Nigam, Matthew Siegler, Robert Stockton, and Takashi Tomokiyo. 2005. Deriving marketing intelligence from online discussion. In *KDD*.
- [53] Carsten Görg, Zhicheng Liu, and John Stasko. 2013. Reflections on the evolution of the jigsaw visual analytics system. *Inf. Visualiz.* (2013).
- [54] John D. Gould and Clayton Lewis. 1985. Designing for usability: Key principles and what designers think. *Commun. ACM* 28, 3 (1985), 300–311.
- [55] Shawn Graham, Scott Weingart, and Ian Milligan. 2012. Getting started with topic modeling and MALLET. Retrieved from <https://programminghistorian.org/en/lessons/topic-modeling-and-mallet>.
- [56] Richard H. Hall and Patrick Hanna. 2004. The impact of web page text-background colour combinations on readability, retention, aesthetics and behavioural intention. *Behav. In. Technol.* 23, 3 (2004), 183–195.
- [57] Abram Handler, Su Lin Blodgett, and Brendan T. O'Connor. 2016. Visualizing textual models with in-text and word-as-pixel highlighting. *ICML Worksh. Hum. Interpret. Mach. Learn.* (2016).
- [58] Abram Handler, Brian Dillon, and Brendan T. O'Connor. 2019. Human acceptability judgements for extractive sentence compression. *ArXiv* (2019). Retrieved from <https://arxiv.org/pdf/1902.00489>.
- [59] Abram Handler and Brendan O'Connor. 2018. Relational summarization for corpus analysis. In *NAACL*.
- [60] Abram Handler and Brendan O'Connor. 2019. Query-focused sentence compression in linear time. In *EMNLP*.
- [61] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. 2002. ThemeRiver: Visualizing thematic changes in large document collections. *IEEE Trans. Visualiz. Comput. Graph.* 8, 1 (Jan. 2002), 9–20.
- [62] Marti A. Hearst. 1995. TileBars: Visualization of term distribution information in full text information access. In *CHI*.
- [63] Jeffrey Heer and Ben Shneiderman. 2012. Interactive dynamics for visual analysis. *Commun. ACM* 55, 4 (2012), 45–54.
- [64] Edward S. Herman and Noam Chomsky. 1988. *Manufacturing Consent: The Political Economy of the Mass Media*. Pantheon Books, New York, NY.
- [65] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*.
- [66] O. Hoeber and Xue Dong Yang. 2006. The visual exploration of web search results using hotmap. In *IV*.
- [67] Jennifer Hoewe and Geri Alumiit Zeldes. 2012. Overturning anti-miscegenation laws: News media coverage of the Lovings' legal case against the State of Virginia. *J. Black Stud.* 43, 4 (2012), 427–443.
- [68] Enamul Hoque and Giuseppe Carenini. 2016. Interactive topic modeling for exploring asynchronous online conversations: Design and evaluation of ConVisIT. *ACM Trans. Interact. Intell. Syst.* 6, 1 (2016).
- [69] Yuening Hu, Jordan Boyd-Graber, and Brianna Satinoff. 2011. Interactive topic modeling. In *ACL*.
- [70] E. Isaacs, K. Damico, S. Ahern, E. Bart, and M. Singhal. 2014. Footprints: A visual search tool that supports discovery and coverage tracking. *IEEE Trans. Visualiz. Comput. Graph.* 20, 12 (2014), 1793–1802.
- [71] Mahmood Jasim, Enamul Hoque, Ali Sarvghad, and Narges Mahyar. 2021. CommunityPulse: Facilitating community input analysis by surfacing hidden insights, reflections, and priorities. In *DIS*. 846–863.
- [72] Mahmood Jasim, Pooya Khaloo, Somin Wadhwa, Amy X. Zhang, Ali Sarvghad, and Narges Mahyar. 2021. CommunityClick: Capturing and reporting community feedback from town halls to improve inclusivity. *Proc. ACM Hum.-comput. Interact.* 4, CSCW3 (2021).

- [73] Hannah Kim, Barry Drake, Alex Endert, and Haesun Park. 2021. ArchiText: Interactive hierarchical topic modeling. *IEEE Trans. Visualiz. Comput. Graph.* 27, 9 (2021).
- [74] Minjeong Kim, Kyeongpil Kang, Deokgun Park, Jaegul Choo, and Niklas Elmqvist. 2017. TopicLens: Efficient multi-level visual topic exploration of large-scale document collections. *IEEE Trans. Visualiz. Comput. Graph.* 23, 1 (2017).
- [75] Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with mechanical turk. In *CHI*.
- [76] Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization - step one: Sentence compression. In *AAAI*.
- [77] Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *EMNLP*.
- [78] Shixia Liu, Michelle X. Zhou, Shimei Pan, Yangqiu Song, Weihong Qian, Weijia Cai, and Xiaoxiao Lian. 2012. TIARA: Interactive, topic-based visual text summarization and analysis. *ACM Trans. Intell. Syst. Technol.* 3, 2 (2012).
- [79] Zhicheng Liu and Jeffrey Heer. 2014. The effects of interactive latency on exploratory visual analysis. *IEEE Trans. Visualiz. Comput. Graph.* (2014).
- [80] H. P. Luhn. 1960. Key word-in-context index for technical literature (KWIC index). *Amer. Docum.* 11, 4 (1960), 288–295.
- [81] Jack Lule. 2002. Myth and terror on the editorial page: The *New York Times* responds to September 11, 2001. *J. Mass Commun. Quart.* 79, 2 (2002), 275–293.
- [82] Dongning Luo, Jing Yang, Milos Krstajic, William Ribarsky, and Daniel Keim. 2012. EventRiver: Visually exploring text collections with temporal references. *IEEE Trans. Visualiz. Comput. Graph.* 18, 1 (2012).
- [83] Trent MacNamara. 2014. Why “race suicide”? Cultural factors in U.S. fertility decline, 1903–1908. *J. Interdiscip. Hist.* 44, 4 (2014), 475–508.
- [84] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.
- [85] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL: System Demonstrations*.
- [86] Adam Marcus, Michael S. Bernstein, Osama Badar, David R. Karger, Samuel Madden, and Robert C. Miller. 2011. Twitinfo: Aggregating and visualizing microblogs for event exploration. In *CHI*.
- [87] Howard Markel, Harvey B. Lipman, J. Alexander Navarro, Alexandra Sloan, Joseph R. Michalsen, Alexandra Minna Stern, and Martin S. Cetron. 2007. Nonpharmaceutical interventions implemented by U.S. cities during the 1918–1919 influenza pandemic. *JAMA* 298, 6 (2007), 644–654.
- [88] Michael Matthews, Pancho Tolchinsky, Roi Blanco, Jordi Atserias, Peter Mika, and Hugo Zaragoza. 2010. Searching through time in the *New York Times*. In *HCIR*.
- [89] Nora McDonald, Sarita Schoenebeck, and Andrea Forte. 2019. Reliability and inter-rater reliability in qualitative research: Norms and guidelines for CSCW and HCI practice. *CSCW* (2019).
- [90] Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *ECIR*.
- [91] Kathleen McKeown and Dragomir R. Radev. 1995. Generating summaries of multiple news articles. In *SIGIR*.
- [92] Kathleen R. McKeown, Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith L. Klavans, Ani Nenkova, Carl Sable, Barry Schiffman, and Sergey Sigelman. 2002. Tracking and summarizing news on a daily basis with Columbia’s newsblaster. In *HLT*.
- [93] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Holberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. 2010. Quantitative analysis of culture using millions of digitized books. *Science* (2010).
- [94] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. Retrieved from <http://arxiv.org/abs/1301.3781>.
- [95] Matthew B. Miles, A. M. Huberman, and Johnny Saldaña. 2014. *Qualitative Data Analysis: A Methods Sourcebook*. (3rd ed.). SAGE.
- [96] John W. Mohr and Petko Bogdanov. 2013. Topic models: What they are and why they matter. *Poetics* 41 (2013), 545–569.
- [97] Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining Text Data*. Springer, 43–76.
- [98] Ani Nenkova and Lucy Vanderwende. 2005. *The Impact of Frequency on Summarization*. Technical Report. Microsoft Research.
- [99] New York Times contributors. 2020. The *New York Times* search page. Retrieved from <https://www.nytimes.com/search>.
- [100] Newspaper.com contributors. 2020. newspapers.com. Retrieved from <https://www.newspapers.com>.
- [101] Janni Nielsen, Torkil Clemmensen, and Carsten Yssing. 2002. Getting access to what goes on in people’s heads? Reflections on the think-aloud technique. In *NordiCHI*.

- [102] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan T. McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *LREC*.
- [103] Seyednaser Nourashrafeddin, Ehsan Sherkat, Rosane Minghim, and Evangelos E. Milios. 2018. A visual approach for interactive keyterm-based clustering. *ACM Trans. Interact. Intell. Syst.* 8, 1 (2018).
- [104] Brendan O'Connor. 2014. MiTextExplorer: Linked brushing and mutual information for exploratory text data analysis. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*.
- [105] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.
- [106] Peter R. Petrucci and Michael Head. 2006. Hurricane Katrina's lexical storm: The use of "refugee as" a label for American citizens. *Australas. J. Amer. Stud.* 25, 2 (2006), 23–39.
- [107] Paul Pierson. 2004. *Politics in Time: History, Institutions, and Social Analysis*. Princeton University Press, Princeton, NJ.
- [108] Peter Pirolli and Stuart Card. 2005. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis*, Vol. 5.
- [109] Proquest LLC. 2020. *Proquest*. Retrieved from <https://search.proquest.com/>.
- [110] Lara Putnam. 2016. The transnational and the text-searchable: Digitized sources and the shadows they cast. *The Histor. Rev.* 121, 2 (2016), 377–402.
- [111] William Revelle. 2020. *Psych: Procedures for Psychological, Psychometric, and Personality Research*. Northwestern University, Evanston, Illinois. R package version 2.0.9.
- [112] Geoffrey Rockwell, Stéfan G. Sinclair, Stan Ruecker, and Peter Organisciak. 2010. Ubiquitous text analysis. *paj J. Initiat. Digit. Human., Media, Cult.* 2, 1 (2010).
- [113] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.
- [114] Evan Sandhaus. 2008. The *New York Times* annotated corpus. *Ling. Data Consort.* LDC2008T19 (2008).
- [115] Benjamin M. Schmidt. 2012. Words alone: Dismantling topic models in the humanities. *J. Digit. Human.* 2, 1 (2012), 49–65.
- [116] Sebastian Schuster and Christopher D. Manning. 2016. Enhanced English universal dependencies: An improved representation for natural language understanding tasks. In *LREC*.
- [117] Carson T. Schütze and Jon Sprouse. 2014. Judgment data. In *Research Methods in Linguistics*, Robert J. Podesva and Devyani Sharma (Eds.). Cambridge University Press.
- [118] Cecilia Di Sciascio, Vedran Sabol, and Eduardo Veas. 2017. Supporting exploratory search with a visual user-driven approach. *ACM Trans. Interact. Intell. Syst.* 7, 4 (2017).
- [119] M. Sedlmair, M. Meyer, and T. Munzner. 2012. Design study methodology: Reflections from the trenches and the stacks. *IEEE Trans. Visualiz. Comput. Graph.* 18, 12 (2012).
- [120] Ehsan Sherkat, Evangelos E. Milios, and Rosane Minghim. 2019. A visual analytics approach for interactive document clustering. *ACM Trans. Interact. Intell. Syst.* 10, 1 (2019).
- [121] Jeannie N. Shinozuka. 2013. Deadly perils: Japanese beetles and the pestilential immigrant, 1920s–1930s. *Amer. Quart.* 65, 4 (2013), 831–852.
- [122] Ben Shneiderman. 1996. The eyes have it: A task by data type taxonomy for information visualizations. In *VL*.
- [123] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. 2016. Expedition: A time-aware exploratory search system designed for scholars. In *SIGIR*.
- [124] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. 2016. History by diversity: Helping historians search news archives. In *CHIR*.
- [125] Janine Solberg. 2012. Googling the archive: Digital tools and the practice of history. *Adv. Hist. Rhet.* 15, 1 (2012).
- [126] John Stasko, Carsten Görg, and Zhicheng Liu. 2008. Jigsaw: Supporting investigative analysis through interactive visualization. *Inf. Visualiz.* 7, 2 (2008), 118–132.
- [127] Jonathan Stray. 2016. What do journalists do with documents? In *Computation+Journalism Symposium*.
- [128] Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel A. Keim, and Oliver Deussen. 2009. Document cards: A top trumps visualization for documents. *IEEE Trans. Visualiz. Comput. Graph.* 15, 6 (2009).
- [129] Nicole Sultanum, Anastasia Bezerianos, and Fanny Chevalier. 2021. Text visualization and close reading for journalism with Storifier. *IEEE Trans. Visualiz. Comput. Graph.* (2021).
- [130] Nicole Sultanum, Michael Brudno, Daniel Wigdor, and Fanny Chevalier. 2018. More text please! Understanding and supporting the use of visualization for clinical text overview. In *CHI*.

- [131] Nicole Sultanum, Devin Singh, Michael Brudno, and Fanny Chevalier. 2019. Doccurate: A curation-based approach for clinical text visualization. *IEEE Trans. Visualiz. Comput. Graph.* 25, 1 (2019).
- [132] Jenifer Tidwell. 1999. A Pattern Language for Human-Computer Interface Design. Retrieved from <https://web.archive.org/web/20201212151556/http://www.mit.edu/jtidwell/commonground.html>.
- [133] Jenifer Tidwell, Charles Brewer, and Aynne Valencia. 2020. *Designing Interfaces* (3rd ed.). O'Reilly Media, Inc.
- [134] Anastasios Tombros and Mark Sanderson. 1998. Advantages of query biased summaries in information retrieval. In *SIGIR*.
- [135] Ted Underwood. 2014. Theorizing research practices we forgot to theorize twenty years ago. *Representations* 127, 1 (2014), 64–72.
- [136] Frank van Ham, Martin Wattenberg, and Fernanda B. Viegas. 2009. Mapping text with phrase nets. *IEEE Trans. Visualiz. Comput. Graph.* 15, 6 (2009).
- [137] Fernanda B. Viégas, Scott Golder, and Judith Donath. 2006. Visualizing email content: Portraying relationships from conversational histories. In *CHI*.
- [138] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. 2000. Guidelines for using multiple views in information visualization. In *AVI*.
- [139] Martin Wattenberg and Fernanda B. Viégas. 2008. The word tree, an interactive visual concordance. *IEEE Trans. Visualiz. Comput. Graph.* 14, 6 (2008).
- [140] Wikipedia contributors. 2021. Robert Mugabe. Retrieved from <https://en.wikipedia.org/wiki/RobertMugabe>.
- [141] Polle T. Zellweger, Bay-Wei Chang, and Jock D. Mackinlay. 1998. Fluid links for informed and incremental link transitions. In *Hypertext*.
- [142] George K. Zipf. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley.

Received July 2021; revised February 2022; accepted March 2022