

AI Mini Chapter: Iteration as Method in Exploratory Data Analysis

Iteration is a core method in digital humanities (DH) research, positioning inquiry as a cyclical process in which writing code, inspecting data, and interpreting visualizations are repeatedly undergone in response to an analyst's emerging insights. With the introduction of generative AI-based tools, iteration in DH extends beyond the cyclical processes of writing code and inspecting visualizations. Large language models (LLMs) can be used to generate content—such as code or suggestions for brainstorming—requiring a new approach to iteration to incorporate critical reflection on prompt refinement, model behavior, and evaluation of AI-generated outputs.

In this section, iteration is introduced as a method in computational humanities research (Tabak 2017). This approach emphasizes using AI to generate content while performing repeated returns to corpus context, showing how computational analysis and close reading work together to inform and reshape interpretation. Iteration will be paired with Exploratory Data Analysis (EDA), a process through which we construct a high-level overview of the corpus, engage in iterative refinement, and draw upon preliminary findings to inform more focused, in-depth investigations.

Up to this point we have written our own code without use of generative AI-based tools like a LLM. Working through the process of writing one's own code step-by-step provides knowledge of the underlying concepts and processes behind working with text as data. Understanding the premise behind code syntax, functions, and how they tie back to data processing enables us to translate research questions into code and actualize our analytic goals. With an established foundation knowledge of data processing we can begin to use generative AI-based tools, like AI chatbots, in a productive and critical way. In this “AI minichapter” we use ChatGPT, though a similar approach could be applied to other AI-based chatbots, including open-source models such as Meta's LLaMA.

Iteration and Exploratory Data Analysis with AI

In practice, exploratory data analysis often begins with diagnostic step: generating simple summaries that reveal dominant patterns, repetitions, and sources of noise in the corpus before any interpretive claims are made. Here, counting words serves a different purpose than in interpretive text analysis. Rather than being used to support historical interpretation, we will think of word counts as functioning diagnostically, helping us understand the attributes of the dataset and informing later decisions about preprocessing and analysis.

While we have already counted words, here it serves a different purpose. As an initial step in exploratory data analysis, we ask a deliberately simple diagnostic question: Which words appear most frequently in the 1850 Hansard debates? By applying minimal preprocessing, we aim to observe the corpus in an unfiltered state and identify dominant structural patterns rather than interpretive themes.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.4      v readr      2.1.6
## v forcats    1.0.1      v stringr    1.6.0
## v ggplot2    4.0.1      v tibble     3.3.0
```

```
## v lubridate 1.9.4      v tidyr      1.3.1
## v purrr      1.1.0
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(tidytext)
library(hansardr)
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
##
## The following object is masked from 'package:dplyr':
##
##      group_rows
```

```
data(hansard_1850)

tokenized_hansard_1850 <- hansard_1850 %>%
  unnest_tokens(word, text) %>%
  count(word, sort = TRUE) %>%
  slice_max(n, n = 20)

tokenized_hansard_1850
```

```
##      word      n
## 1    the 2588864
## 2     of 1404831
## 3    to 1126226
## 4   that  726686
## 5    and  706117
## 6     in  597753
## 7     a  522476
## 8    was  454102
## 9     he  454073
## 10    it  359726
## 11    be  351663
## 12   had  325190
## 13 which  297801
## 14   for  282983
## 15   not  274629
## 16 would  230866
## 17    as  210520
## 18   by  208331
## 19 with  194136
## 20 this  187449
```

At this exploratory stage, a raw word count is not intended to produce interpretation. Instead, it exposes a central characteristic of the corpus: the abundance of articles and function words. This diagnostic result signals the need for preprocessing steps—such as stop-word removal—to clarify patterns relevant to historical inquiry.

```
data(stop_words)

tokenized_hansard_1850 <- hansard_1850 %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by = "word") %>%
  count(word, sort = TRUE) %>%
  slice_max(n, n = 20)

tokenized_hansard_1850
```

```
##      word      n
## 1      hon 125465
## 2     house 118861
## 3 government 95417
## 4      bill 77322
## 5     noble 73180
## 6      lord 68942
## 7   country 64993
## 8   question 55827
## 9 gentleman 53800
## 10     time 47893
## 11       000 47519
## 12   subject 42536
## 13 committee 40020
## 14       law 38573
## 15    friend 35456
## 16   opinion 32346
## 17    public 31294
## 18       sir 30859
## 19 parliament 28816
## 20    measure 28536
```

We can also use AI chatbots in this process by facilitating brainstorming and supporting iterative analysis through the automation of tasks. For instance, while we already removed stop words (e.g., “the” or “and”) from our dataset, which significantly improved the clarity of our results. We can also use ChatGPT to generate additional stop words, specifically attuned to the linguistic characteristics of 19th-century Hansard.

At this exploratory stage, patterns in the data raise new questions rather than provide answers. The prominence of procedural and honorific language suggests that additional preprocessing may be necessary to clarify topical signals. Rather than relying on a fixed or generic stop-word list, we can treat this decision itself as part of the exploratory process.

Exploratory Exercise: Proposing Corpus-Specific Stop Words

Here we use ChatGPT for brainstorming during exploratory data analysis, not to produce authoritative responses, but to suggest stop words that can then be evaluated, refined, or rejected through our knowledge of the corpus and by reflecting on our research direction.

Analyst Prompt

Generate a list of stopwords to be removed for an analysis of the 19th-century Hansard corpus.

ChatGPT-4o Response

```
custom_stopwords <- c("lord", "lords", "hon", "mr", "sir", "gentleman", "gentlemen",  
  "speaker", "right", "honourable", "house", "parliament",  
  "minister", "bill", "member", "question", "motion", "committee",  
  "government", "debate", "order", "division", "clause")
```

Before removing the stop words, we can think critically about whether or not the generated results align with our own analytic goals. For example, the word “government” appears frequently in parliamentary debate and may function as procedural language in some contexts that can be safely removed for analysis. However, if the research question involves examining how political authority is framed, contested, or attributed, retaining this word becomes necessary in order to analyze its collocates and surrounding rhetorical patterns.

This reflective step reinforces an important principle of exploratory data analysis: preprocessing decisions are provisional and question-dependent rather than fixed or universal. Words that operate as noise in one analysis may serve as signal in another. AI-generated stop-word lists should therefore be treated as hypotheses to be tested through inspection, rather than rules to be applied automatically.

In practice, this means iteratively revisiting preprocessing choices like stop words removal as analytic focus sharpens. A researcher may begin by removing a broad set of procedural terms, then selectively reintroduce specific words—such as government—to examine their contextual usage, collocates, or shifts in meaning across debates. This back-and-forth movement between diagnostic exploration and focused analysis exemplifies how exploratory data analysis supports interpretive work in digital history.

Exploratory Exercise: From Overview to Focus

Building on the exploratory data analysis practices introduced above, this exercise demonstrates another way AI chatbots can support analytical workflows during the early stages of research by suggesting topics to further explore. As with diagnostic word counts, the goal here is not interpretation but orientation and exploration of the data. We use AI as a brainstorming tool to help scope potential areas of inquiry by proposing candidate words for further investigation. These suggestions serve as provisional starting points that guide subsequent analysis, rather than as definitive claims about historical significance.

User Prompt

I want to text mine and analyze heated topics from within the 1850 Hansard Parliamentary debates. Suggest several keywords that might be interesting.

ChatGPT-4o Response

When analyzing the 1850 Hansard Parliamentary debates for heated topics, you'll want to focus on issues that were contentious or widely discussed at the time. Here are some keyword suggestions:

Corn Laws – Although the Corn Laws were repealed in 1846, the economic and social impacts continued to be a hot topic of debate.

Irish Famine – The Great Irish Famine (1845–1852) was a significant issue, particularly regarding the British government's response.

Chartism – Though the Chartist movement had declined by 1850, discussions about labor rights, suffrage, and political reform were still prevalent.

Poor Law – The administration and reform of the Poor Laws were highly debated, particularly in relation to poverty and public welfare.

Colonial Governance – Issues around the governance and management of British colonies, particularly in places like India and Ireland.

Free Trade – Discussions around free trade policies, especially in the aftermath of the Corn Laws repeal.

Slavery – Although the British Empire abolished slavery in 1833, discussions about slavery and the slave trade (particularly in relation to other nations) continued.

Catholic Emancipation – Ongoing debates about the rights of Catholics in Britain and Ireland.

Industrialization – Issues related to the rapid industrialization of Britain, including worker rights, factory conditions, and urbanization.

Public Health – Particularly in the context of the cholera outbreaks and the general health of the urban poor.

These keywords should help you identify debates that were particularly heated or significant during that time. You can further refine your analysis by focusing on specific phrases or related terms within these broader topics.

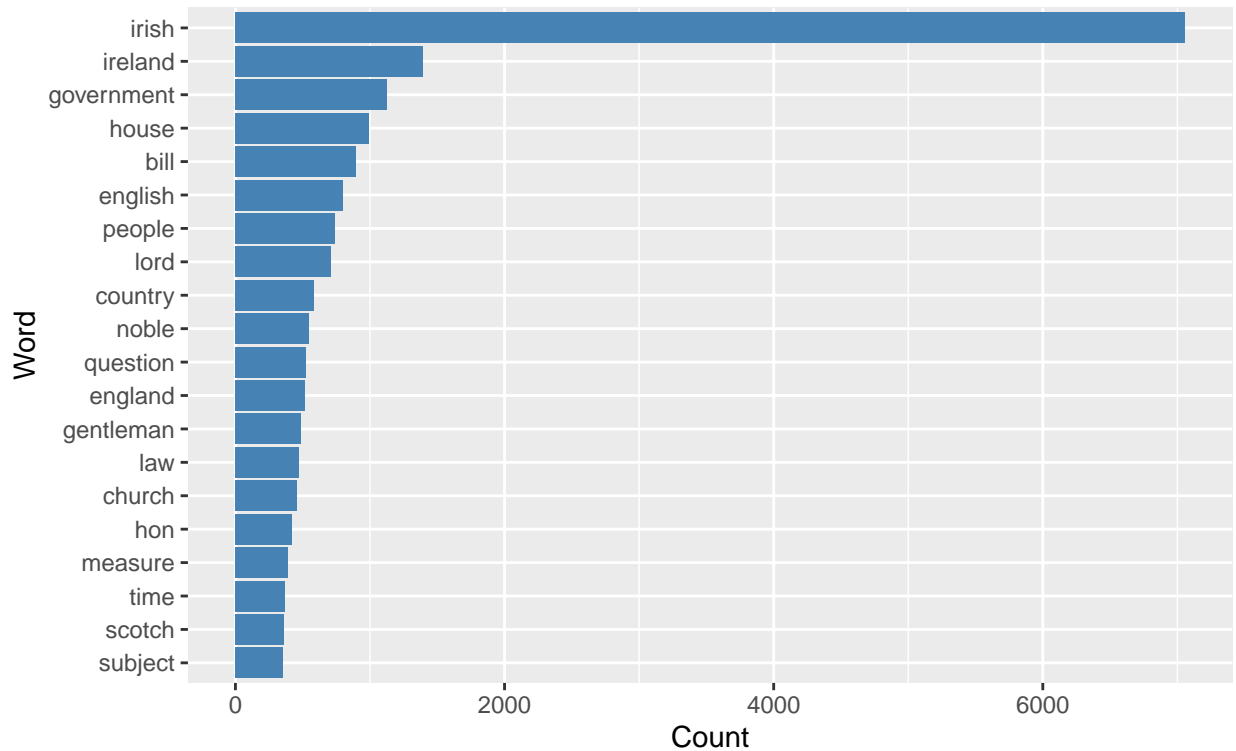
This response offers a high-level overview of potentially interesting topics within the corpus. Treating these themes as candidate keywords allows us to move from exploratory scoping to more focused analysis. To demonstrate this transition, we use the code introduced earlier in an earlier chapter to examine the lexical context surrounding the term Irish in the 1850 Hansard debates.

```
irish_context <- hansard_1850 %>%
  filter(str_detect(text, "\\bIrish\\b")) %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  filter(!str_detect(word, "[:digit:]")) %>%
  count(word) %>%
  arrange(desc(n))
```

```
top_irish <- irish_context %>%
  top_n(20)

ggplot(data = top_irish) +
  geom_col(aes(x = reorder(word, n),
    y = n),
    fill = "steel blue") +
  coord_flip() +
  labs(title = "Top Words Occuring in Sentences Mentioning Irish",
    subtitle = "From the 1850 Hansard debates",
    x = "Word",
    y = "Count")
```

Top Words Occuring in Sentences Mentioning Irish From the 1850 Hansard debates



[Needs more analysis – do students have feedback?]

Exploratory Exercise: Brainstorming Historical Events

We can use AI chatbots to assist with identifying additional historical events for analysis by working within a controlled vocabulary. Rather than generating open-ended interpretations, the chatbot is guided to propose candidate events that align with previously defined analytic criteria. This approach supports iterative exploration while ensuring that the scope of inquiry remains grounded in the analyst's existing interests.

Here, we prompt the chatbot to suggest events that may be relevant to the study of war-related language in the year 1840. To constrain and focus the response, we provide examples drawn from the an eventslist dataset, which functions as a reference vocabulary for the types of events of interest.

```
data(hansard_1840)
data("events")

eventslist <- events %>%
distinct(event_name, scholar_assigned_date) %>%
filter(!scholar_assigned_date > 1840) %>%
mutate(event_name = tolower(event_name)) %>%
select(event_name, scholar_assigned_date)

head(eventslist)
```

```
## # A tibble: 6 x 2
##   event_name      scholar_assigned_date
```

##	<chr>	<dbl>
##	1 french revolution	1789
##	2 magna carta	1215
##	3 norman conquest	1066
##	4 corn laws	1815
##	5 battle of boyne	1690
##	6 glorious revolution	1688

To structure working with ChatGPT, we provide it with a template specifying the required output format. Providing this constraint in the prompt helps standardize the output, making it easier to inspect.

Analyst Prompt

I am providing you with a list of events followed by their year. The purpose of this list is to analyze key events that were mentioned in the 1860 Hansard debates. Name some different events that might be relevant based on the list I provide you. Don't mention the same as provided by the list. Also provide a reason for why the events are relevant.

Return your answer in this format:

Event:

Reason:

These are the events from the list: (copy/paste contents of eventslist)

ChatGPT-4o Response

Event: Abolition of the Slave Trade (1807)

Reason: The abolition of the slave trade would be highly relevant in discussions about moral and economic reforms during the 1860 Hansard debates, particularly in the context of human rights and British colonial policies.

Event: Reform Act (1832)

Reason: This act, which expanded suffrage and restructured parliamentary representation, would likely influence discussions on electoral reform and representation in 1860.

Event: Crimean War (1853–1856)

Reason: The Crimean War had significant impacts on British foreign policy, military reforms, and public health improvements, all of which could be relevant topics of debate in 1860.

Event: Irish Potato Famine (1845–1852)

Reason: The famine's aftermath would remain a pressing issue, influencing debates about Irish governance, economic policies, and relief efforts.

Event: Great Exhibition (1851)

Reason: As a symbol of industrial progress and imperial power, the Great Exhibition could be discussed in terms of economic innovation and Britain's global role in the mid-19th century.

Event: Indian Rebellion (1857)

Reason: The rebellion was a turning point in British imperial governance, leading to significant debates about military strategy, colonial administration, and the justification of imperial rule.

Event: Public Health Act (1848)

Reason: This act, which addressed urban sanitation and public health, would be relevant to discussions about improving living conditions and public health policies in 1860.

Event: Establishment of the Penny Post (1840)

Reason: The reform in postal services, increasing communication efficiency and accessibility, could relate to debates on economic and infrastructural development.

Our initial prompt to the AI chatbot successfully guided it to identify and return events similar to the query. However, when we provided the chatbot with the entire contents of eventslist, the results lacked precision and specificity. This outcome suggests that more targeted guidance is necessary to focus on the specific characteristics of interest. To tailor the results to a specific research question, we will refine the

prompt further by narrowing in on the events we are most interested in from `eventslist` and instructing the chatbot to return like-events, in this case, other names of meetings that occurred in the 1860s.

Analyst Prompt

I am providing you with a list of events followed by their year. The purpose of this list is to analyze key events that were mentioned in the 1860 Hansard debates. Name some different events that might be relevant based on the list I provide you. Don't mention the same as provided by the list. Also provide a reason for why the events are relevant based on the list I provide.

Return your answer like this:

Event:

Date:

Reason:

These are the events from the list: "greek war", "napoleonic wars."

ChatGPT-4o Response

Here are some different events that might be relevant based on your list:

Event: Chartist Movements

AI Assigned Date: 1838–1850

Reason: The Chartist movements, with their organized public gatherings for political reform, directly relate to the mentioned events like "public meeting" and "recent meeting," illustrating how assemblies have historically been used to drive societal change.

Event: Anti-Corn Law League Meetings

AI Assigned Date: 1839–1846

Reason: These meetings share a thematic connection to "public meeting" by demonstrating how public gatherings have been pivotal in shaping discourse around economic reform, similar to those discussed in the 1860 debates.

Event: Hyde Park Demonstrations

AI Assigned Date: 1855–1860

Reason: Hyde Park demonstrations resonate with "public meeting" and "recent meeting," showing how such gatherings functioned as platforms for political and social advocacy, reflecting the period's reliance on public discourse for progress.

Event: Meetings on the Reform Act of 1832

AI Assigned Date: 1831–1832

Reason: These meetings are akin to "yorkshire meeting" and "clontarf meeting," showcasing historical instances where regional or national public assemblies catalyzed significant legislative reforms, serving as a possible reference point in 1860.

Event: Fenian Meetings

AI Assigned Date: 1858–1860

Reason: While slightly later, Fenian meetings parallel "clontarf meeting" by highlighting Irish political concerns and illustrating how public gatherings were used to address issues of representation and governance.

Now that we have a more specific list of events that aligns with our interests, we can use them in our analysis. To use our new events, we will create a data frame for them. We can prompt the chatbot to return the needed code. Given that we are very familiar with creating data frames at this point, we can also proof the chatbot's results and ensure they match our desired output.

Analyst Prompt

Return this text as a data frame in R with two columns: a column for the event, named `event_name` and a column for the date, named `AI Assigned Date`. Write the events in lower case.

ChatGPT-4o Response

Here is the R code to create the requested data frame:

```
# Create the data frame

events_df <- data.frame(
  event_name = c(
    "crimean war",
    "russo-turkish war",
    "congress of vienna",
    "siege of acre",
    "battle of navarino",
    "schleswig-holstein wars"
  ),
  AI_Assigned_Date = c(
    "1853-1856",
    "1828-1829",
    "1814-1815",
    "1799",
    "1827",
    "1848-1852"
  )
)

# Print the data frame
print(events_data)
```

This will create a data frame in R with two columns: event_name (containing the event names in lowercase) and AI Assigned Date.

We can now test the chatbot's responses by running the code.

```
# Create the data frame
events_df <- data.frame(
  event_name = c(
    "crimean war",
    "russo-turkish war",
    "congress of vienna",
    "siege of acre",
    "battle of navarino",
    "schleswig-holstein wars"
  ),
  AI_Assigned_Date = c(
    "1853-1856",
    "1828-1829",
    "1814-1815",
    "1799",
    "1827",
    "1848-1852"
  )
)

# Print the data frame
```

Table 1: Excerpt from Hansard 1850

sentence_id	text	speechdate	debate	matched_event
S3V0156P0_3555	As to the noble Earl's argument against the abandonment of a prohibitive duty on coal in the event of war, I have only to say that at the time of the Crimean war we did issue a proclamation prohibiting the export of contraband of war to countries north of Dantzic, but that we did not extend it to coal.	1860-02-20	QUESTION.	Crimean war
S3V0156P0_8154	The Crimean war, while it created an enormous extra demand for ships, curtailed in a very small degree the usual channels of commerce, and the consequence was that a larger number of ships was called into requisition than was necessary for the usual trade of the country.	1860-01-31	COMMITTEE MOVED FOR.	Crimean war
S3V0156P0_8400	Member for Sunderland himself, who, in a work which he had recently published, made mention of the adversity under which the shipping interest had laboured over since the date of the Crimean war, and declared the prediction that that adversity was only a mere passing cloud to have been completely falsified.	1860-01-31	COMMITTEE MOVED FOR.	Crimean war
S3V0156P0_10788	With regard to the claims of officers of the Land Transport Corps, a Committee of that House had sat, before which was brought the case of certain officers who had entered that corps under very peculiar circumstances, and who at the conclusion of the Crimean War were dismissed from that service.	1860-02-03	OBSERVATIONS.	Crimean War
S3V0156P0_14522	He did not think it possible to conceive waste like that which had taken place in disposing of the surplus stores after the Crimean war.	1860-02-10	COMMITTEE.	Crimean war

```
print(events_df)
```

```
##           event_name AI_Assigned_Date
## 1         crimean war       1853-1856
## 2      russo-turkish war       1828-1829
## 3    congress of vienna       1814-1815
## 4         siege of acre         1799
## 5    battle of navarino         1827
## 6 schleswig-holstein wars       1848-1852
```

```
data("hansard_1860")
data("debate_metadata_1860")

hansard_1860 <- hansard_1860 %>%
  left_join(debate_metadata_1860)
```

```
## Joining with 'by = join_by(sentence_id)'
```

The following code filters the `hansard_1860` dataset to include rows where the `text` column matches any event name in `events_df$event_name`, using case-insensitive matching. It then adds a new column, `matched_event`, which contains the specific matched event name from the `text`. The result is a filtered data frame with an additional column identifying the matched event for each row.

```
# Match with each event and
# add a column to capture the matched event
new_matched_events <- hansard_1860 %>%
  filter(str_detect(text,
    regex(paste0("\\b(", paste(events_df$event_name, collapse = "|"), ")\\b"),
      ignore_case = TRUE))) %>%
  mutate(matched_event = str_extract(text,
    regex(paste0("\\b(", paste(events_df$event_name, collapse = "|"),
      ignore_case = TRUE))))
```

The results show that not all of the events were mentioned word-for-word in the Hansard corpus; however, we successfully matched several, including the Crimean War, Congress of Vienna, and Battle of Navarino.

While an AI chatbot can suggest a wider range of related terms to search for—and these terms might spark our curiosity—those terms may not actually appear in the corpus. For experts 19th-century history, it might be expected that some phrases would not be found, but for others, this process can offer a useful way to explore which events were and were not mentioned in the data. This kind of exploration is part of an ongoing, iterative process—where analysts refine their searches by adding or adjusting keywords, either manually or with AI.

It is also important to note that the AI chatbot did not automatically fix formatting issues in the text, such as inconsistent capitalization. To obtain more reliable and visually meaningful results, the analyst may want to take extra processing steps to clean the data—such as making all text lowercase and fixing other formatting problems—before analysis.

```
event_counts <- new_matched_events %>%  
  count(matched_event, name = "count") %>%  
  arrange(desc(count))  
  
print(event_counts)
```

```
##      matched_event count  
## 1      Crimean war   360  
## 2      Crimean War   143  
## 3 Congress of Vienna    42  
## 4 battle of Navarino     5
```

Conclusion

In this chapter, we operationalized iteration as a core method in digital humanities research by integrating historical theory, code, exploratory data analysis, and artificial intelligence into a single, cyclical workflow. Rather than treating AI as a replacement for scholarly interpretation, we positioned generative AI-based tools as participants in an iterative process that requires repeated inspection of data, returns to corpus context, and critical evaluation of AI-generated outputs.

By pairing iteration with exploratory data analysis, we demonstrated how high-level diagnostic exploration can inform subsequent analytic decisions, and how provisional findings can guide more focused investigations. Throughout the chapter, AI was used to support this process—assisting with brainstorming, proposing preprocessing strategies, and accelerating iteration—while interpretive judgment remained grounded in close reading and historical expertise.

The chapters that follow build on this foundation by introducing more advanced computational techniques and by examining, in greater depth, how generative AI reshapes iterative research practices in the humanities. A central question we pursue is whether—and under what conditions—current AI chatbots can meaningfully contribute to the interpretation of nineteenth-century Parliamentary debates, and how scholars might critically assess those contributions.

While generative AI offers tools for exploratory analysis and iteration, it also introduces new methodological and epistemic risks. Accordingly, we will continue to examine the limitations of AI-assisted approaches to computational text analysis, paying particular attention to the moments when AI-generated suggestions must be interrogated, revised, or rejected in order to sustain historically grounded and responsible interpretation.

New citations for cabot:

Tabak, Edin. 2017. “A Hybrid Model for Managing Digital Humanities Projects.” *Digital Humanities Quarterly* 11 (1). Roopika risam new digital worlds