Carnegie Mellon University
Human Computer Interaction Institute

PhD Thesis Proposal

Supporting Sensemaking through Data Flows

by

Nathan Hahn

Advisor: **Aniket Kittur**
Adam Perer
Brad Myers
Jaime Teevan

September 4th 2019

Abstract

Supporting Sensemaking through Data Flows

The internet has become the de-facto information source for individuals — from finding a new cookie recipe, to learning how a transistor works. Tools for helping users find answers to their questions have grown tremendously in response; a user can get an answer in their search results for when the next Pirates game is, or get a list of facts about their favorite movie actress. However, for many questions, such as buying a car, there isn't one right answer — the best choice for an individual depends on their particular set of circumstances and personal preference. For these situations, it's up to the user to make sense of the answer space: accumulate what options are available, what the differences and features of these options are, and eventually choose between them.

This process, sensemaking, is a highly iterative and cyclical process, where information is constantly being found, incorporated, restructured, summarized, and generalized. As users continue to collect new information, they need to adjust and restructure their existing information, while incorporating the key known points of the new information into their understanding of the problem. This constant adjustment of both the data and structure surrounding the data puts a significant mental burden on users, and often requires them to resort to external means to track and manage this information. In the context of online sensemaking, this can be done in notepads, tabs, word prcessors, spreadsheets, kanban boards, or even emails. As users proceed to move along with their data in this process, they need to manually update and transfer data between these tools, which might often be more trouble than its worth.

In this work, I explore interactive systems which provide support for the transitions between phases of the process, or dataflows. I theorize that tools which are able to support the underlying mechanisms occurring during these transitions, through either computation or interaction, will decrease the cognitive load on users and allow for support of other sensemaking requirements, such as easy resumption and refinding. My previous work in using crowdworkers to answer complex questions suggests that a reasonable light-weight scaffolding exists for supporting the sensemaking process, as crowdworkers in that system were able to produce a reasonable output while spending less than 5 minutes on a single task. Three of my other systems, Bento, Siphon and Distil, are prototype systems designed at naturally supporting collecting sources, extracting information, and organizing collected content. In my proposed work, I will explore the portion of this workflow: evaluating and decision making with the structured information.

To perform this exploration, I plan to build on top of a system we have been

developing, Fuse, that is a refined version of my previous systems for source management, content collection and organization. With Fuse, I will develop and test different interfaces for generating decision-making support artifacts for sensemaking. Several different interface forms, such as tables, mind maps, lists and affinity diagrams, are already utilized by users performing sensemaking, however the cost of generating and adding content to these diagrams can create a substantial mental burden on the user, slowing or even preventing their adoption. Understanding how to more fluidly generate these forms will be a key piece of this proposed work. I plan to evaluate these interfaces with a combination of controlled user studies, as well as field deployment of them to users of the Fuse system.

Contents

Chapter 1

Introduction

People are increasingly relying on web-based information sources to make sense of unfamiliar domains. With the ever increasing breadth, depth and diversity of online information sources, individuals are using the internet for a wide range of research tasks, such as understanding medical diagnosis [38], performing in-depth product comparisons [125], or creating an itinerary for an upcoming trip [22]. While some online information tasks have simple, uncomplicated answers, such as the score of the Steelers football game, others, such as planning a week long vacation, can vary significantly based on situational factors (time of year, age of individuals on the trip, etc.) [30, 104]. In order to understand how the information they encounter online interacts with their personal situation, users engage in the process of **sensemaking**. Through this process, users collect and develop an information landscape around a particular topic, allowing them to make decisions, answer questions, or generate hypotheses [130].

In its simplest and most general form, sensemaking is the process of combining existing and current information with situation specific parameters to achieve some sort of goal [43]. As researchers have begun to investigate this process in greater detail [84, 123, 130, 159], they have uncovered the unique role that structure development plays in this process. As users come to understand a particular information space, they develop a structure, or representation of the space [130], which they then utilize to seek out additional information and further refine their understanding, or they eventually leverage for their goals. This structure is developed and utilized in both a bottom-up, as well as a top-down manner, with users typically going through several cycles of gathering, summarization and refinement before they have a finalized structure [123, 130, 159]. In Priolli and Card's notional model [124], they divide up the process into two additional loops - an "information foraging" loop, where individuals are collecting sources of information to exploit, and then the "sensemaking" loop, where problem structuring, evidentiary reasoning and decision making take place. These two processes are tightly coupled together, and one drives the action in the other. As noted in Klein's data frame sensemaking model, there is a constant push and pull between trying to fit data found into a particular structure, as well as adjusting and refining that structure to appropriately contain the data.

For example, imagine an individual shopping for a camera for the first time. Based on their previous interactions with cameras of their friends, they can understand that there are different sizes of cameras, cameras with different levels of zoom, as well as

different output quality, and price levels. Because they plan on traveling a lot with the camera, they'd prefer for it to not be too bulky, but still take excellent pictures. This starting frame, or structure, is then utilized for driving their initial search – they might look for "best cameras for travel". As they come across potentially good models, data in this case, they might write them down in a document for further exploration or price comparison across different websites, along with notes from the sources saying why those are good cameras. As they begin to dig deeper into the data, they discover that quality is more complicated – there are cameras that perform well in low light, have better color accuracy, or are higher resolution. Additionally, there are often tradeoffs between camera size and zoom level, as well as price and customizability. Using this additional knowledge, they can update their frame / structure, and then revise their searching strategy to look for a "full frame compact camera," as that would fit their quality and size constraints the best. They then have to go back to their existing list of cameras and update them, saying which ones are full frame, which ones perform well in low light, etc. or find additional data to fill those gaps. This process will continue until the searcher feels like they have enough information to make a decision and purchase a camera.

1.1 Online Sensemaking

To perform this process online, a user might use a number of different tools, interfaces, and information intermediaries to help them. First, they would most likely use a search engine to look up different cameras, or answer questions about camera features. They might then collect different useful, or potentially useful pieces of information as a collection of tabs, screenshots from pages, excerpts, or links. As they start to assemble a working list of cameras they want to consider, they then leverage a document or a spreadsheet to organize their findings into. Lastly, as they come across incongruent information, missing information, etc. they insert placeholders into their artifact, or generate a list of todo items that they need to look out for as they continue their searching process. Uniquely, due to trends in interface usability, enhanced methods for input, computing power, and device ubility almost all of this process can, and might, occur on a single computing device [112, 135, 136]. Previously, due to the spread and availability of online information, as well as the rigidity of user interfaces, users might resort to physical media such as books, pen and paper or post-it notes to perform a significant portion of their sensemaking process [2, 123, 130, 137].

The ability to perform the sensemaking process in its entirety on a computing device offers a unique opportunity to take advantage of some key features of modern computing: ubiquity of information access, easy distribution / collaboration, information data flows, and high speed data processing. While the community has begun to explore some of these facets through collaborative sensemaking interfaces [77, 82, 111, 113, 114, 119, 120, 150], enhanced foraging interfaces [32, 39, 89, 143], and unique interfaces for presentation and organization [59, 70, 98], most of these tools exist as standalone tools for just one portion of the process, limiting their ability to support the user as they transition from one phase to the next. In this work, I focus on supporting those transitions by framing them as a series of **information data flows**: the transfer of information from one

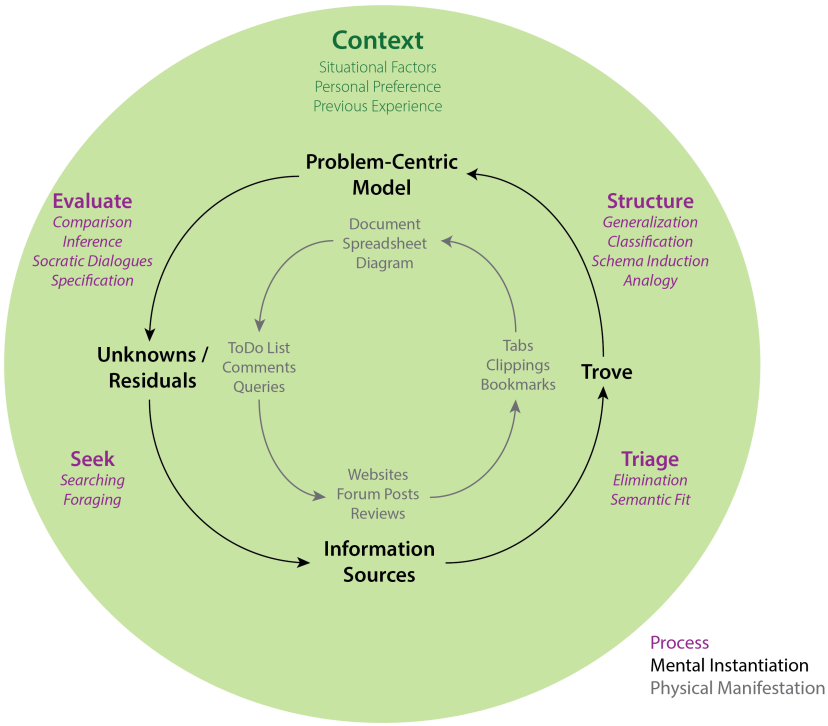


Figure 1.1: A dataflow-centric sensemaking workflow model

thing to another. In this instance, it's the transfer of information from one phase of the sensemaking process to the next, and how information can be summarized, compacted, re-found, and resurfaced at the appropriate moment for the user.

I posit that **interactive tools can serve as a more effective means of bridging the different phases of the sensemaking process by creating data flows that support the underlying cognitive mechanisms**. By focusing on this transfer and transformation of data, users can experience lower cognitive load and overhead when performing sensemaking on computing devices. Additionally, due to the lower overhead and ease of expression, users can externalize their internal process more efficiently, allowing them to more easily suspend their process, collaborate with others, gain receive computational support. The following framework bridges cognitive sensemaking models [124, 130, 159], with the cognitive mechanisms [159] and digital tools utilized during the sensemaking process [22, 49, 104, 136, 146]. In this thesis I present a set of tools that work to support a transition from one or more phases to the next by providing an interactive data flow following one of the higher level cognitive mechanisms.

1.2 Overview

In the first part of this thesis, I present the Knowledge Accelerator (KA): a system for distributing the sensemaking process across a large group of individuals using microtasks **(Chapter 3)**. This system allowed us to understand what a reasonable basic workflow looks like for individuals performing sensemaking, the limitations of which information needed to be "global" – or spread across the entire process, and what information needed

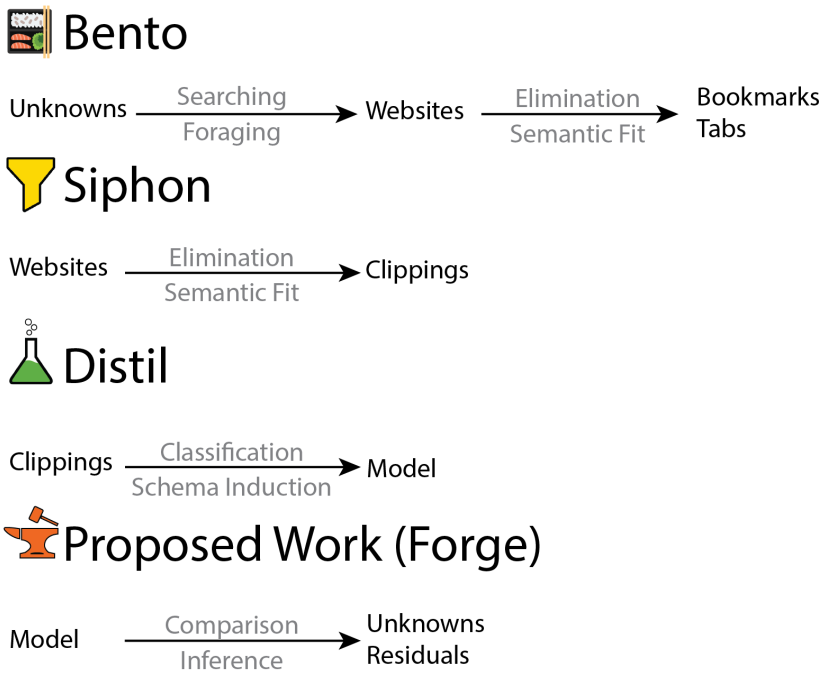


Figure 1.2: A breakdown of the individual proposal projects based on the dataflow-centric model

to be conveyed from one step of the process to the next. Due to the artificial constraint we imposed on ourselves of doing this process entirely through microtasks, we were able to test what the minimum required information transfer needed to occur between the different stages, as well as which stages offered the most potential for machine support.

Using the information gained from the KA system, I developed Bento: a tool for source management **(Chapter 4)**. Bento was designed to tackle the first part of the first part of the sensemaking process: moving from one or more unknowns to a series of possible information sources to a curated final set of sources. By creating a "sensemaking workspace", similar to a task-based desktop or workspace [48, 147], users can record their list of current unknowns by creating persistent queries that live along with their workspace. These queries are then executed as searches, through which users can triage the results through trashing, starring, and gaining progress indicators on the list of results. Through these features Bento was able to make users feel more organized and resume their sensemaking activities more rapidly compared to a traditional web browser.

While Bento gave users the ability to triage information at the source level, users are unable to effectively break apart and triage sections of an information source. Throughout the sensemaking process, users have varying levels of uncertainty [28], resulting in them often selecting large sections of information early on in the process, and smaller, key sections later. To tackle this issue, I developed the Siphon toolkit, which allows users to use a variety of selection interactions to annotate and extract information ranging from an entire page, to a specific word **(Chapter 5)**. Uniquely, Siphon also maps any selection to the underlying HTML of the webpage, which allows the toolkit to pass along the underlying content, highlight and maintain a connection

to the selected content, and rerender the selection in other contexts.

With tools built that allow users to find and triage information efficiently, I then focused on assisting users with generating a model. Because Sensemaking is a cyclical process, often with fairly quick iteration, a user's model, data sources, and triaged information change rapidly. Manual or cluster-based organizations might work for one instance of the cycle, but could become quickly outdated the next, requiring significant effort on the user's part to reorganize what they've collected. I developed Distil to assist users with this model generation challenge: through interactive "smart categories", users can define auto-updating categorizations that automatically pull in relevant existing and new information **(Chapter 6)**. These smart categories allow users to more efficiently perform the processes of classification and schema induction on their collected data through streamlined categorization. With Distil, users were able to quickly create and adjust their categorizations, using them to both more deeply explore the dataset as well as organize it.

Through the above tools, I was able to streamline the digital sensemaking experience from query to model. By providing an enhanced set of tools through which users could perform the processes of seeking, triaging and structuring, users are able to more effectively transfer information from the beginning of their sensemaking process to later stages on computing devices with lower interaction costs. For my proposed work, I plan to take the underlying concepts of my previous tools and combine them into a single tool: Fuse **(Chapter 7)**. I will then deploy Fuse to a large group of users to test the combined effects of a streamlined digital dataflow for sensemaking. Lastly, I plan to explore the last portion of the dataflow-centric sensemaking workflow, evaluation, through additional alternative data structures more suited for the operations of comparison and inference, such as a table or an affinity diagram.

Chapter 2

Background

First to provide some context surrounding online information seeking tasks and current user tools, I will provide a high level overview of information seeking and sensemaking behavioral models. I will then connect these to the dataflow-centric model created in this proposal, and use this to discuss some of the previous sensemaking support systems.

2.1 Sensemaking Models

Sensemaking is generally considered to be an iterative process where a user is building up an understanding of an information space in order to achieve a goal [43, 130]. Theories and related empirical work point out that unlike simple factual information finding (e.g., what is the weather, when was someone born), for complex sensemaking tasks like shopping or making health decisions finding relevant information sources is only the first step in the search process [130, 152]. Users must also perform additional synthesizing to produce an actual understanding. A number of models of sensemaking have been proposed, including Russell et al.'s cost structure view [130], Dervin's sensemaking methodology [42], Klein et al.'s data-frame model [83], organizational process views [53], organizational adaptation views [40, 109], the notional model by Pirolli and Card [124], and the comprehensive model by Zhang et al. [159].

In this work, I propose an additional model: the dataflow-centric model of sensemaking. This model draws primarily from Pirolli and Card's notional model [124] and Zhang's comprehensive model [159]. The notional model, developed through cognitive task analysis, defines ten processes and six representations of the sensemaking process (Figure 2.1). These representations are presented in a waterfall where a user utilizes bottom-up processes to move to a higher level representation of the data, and top-down processes to evaluate and fill gaps in the representation. Generally, this process moves from information, to schema, to insight, and finally product, with two large loops of activity:

- An information foraging loop where the sensemaker is finding, filtering, reading and extracting information
- A sensemaking loop where the sensemaker is iteratively building and refining a mental model that best explains the data

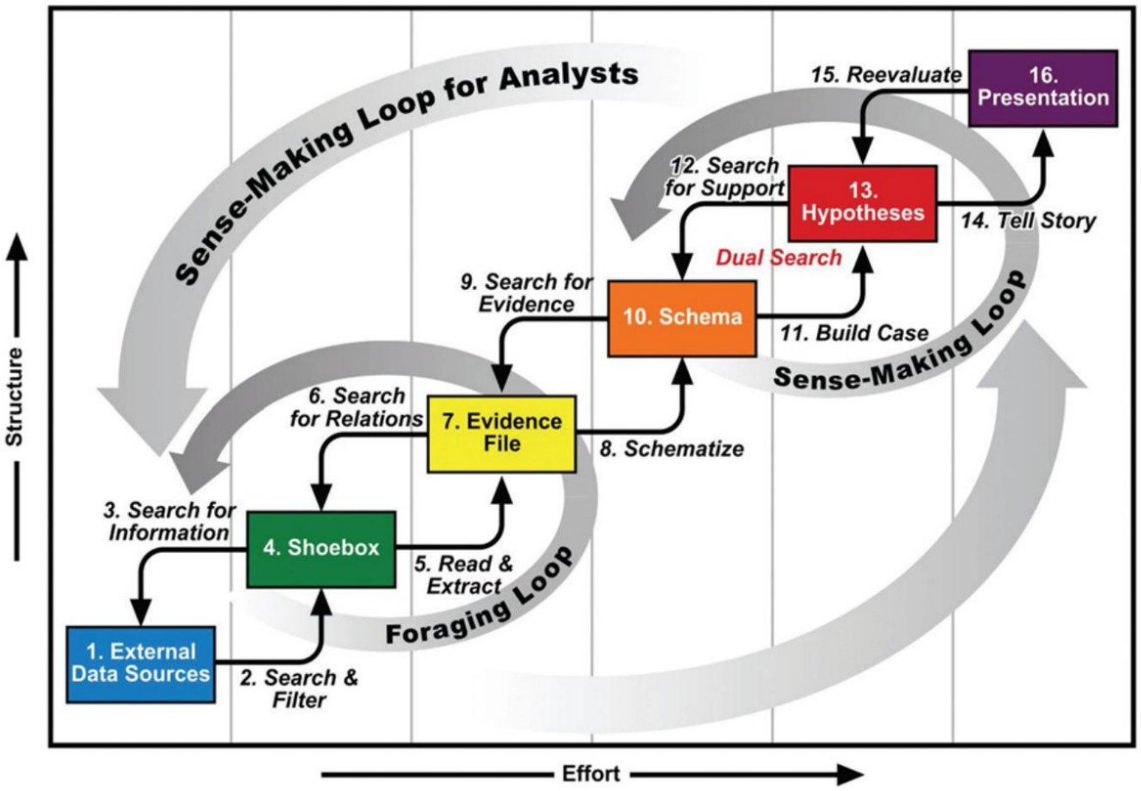


Figure 2.1: Pirolli and Card's notional model of sensemaking

Zhang et al. adjusts this model by suggesting specific ways in which the structure might be adjusted over time or in one loop, as well as the role that external representations play in the schema generation process (Figure 2.2) [159]. Additionally, Zhang defines a set of top-down and bottom-up cognitive processes from literature in reasoning, reading comprehension and learning that operate on the structure loop and data loops (Figure 2.3).

The dataflow-centric model defines three key segmentations of the sensemaking process: a series of cognitive processes occurring between the steps, a set of mental instantiations of the intermediates of the process, and a set of physical artifacts produced as intermediaries in the context of online / digital sensemaking. The outer cognitive processes are taken from Zhang's list of top-down and bottom-up mechanisms, which map loosely to the set proposed by the notional sensemaking model. The top-down mechanisms are spread between the "Evaluate" and "Triage" actions, as these are the stages where users are leveraging their structure or model to identity, filter, and fill information gaps. The bottom-up mechanisms are contained in the "Structure" process, as this is where users take the residual information from their triage process that isn't able to be appropriately fit to adjust and update their structure [130]. The mental instantiations are an abreviated set of the notional model's six representations:

- Information Sources: These are equivalent to the external data sources
- Trove: A combination of the "shoebox" and "evidence file"

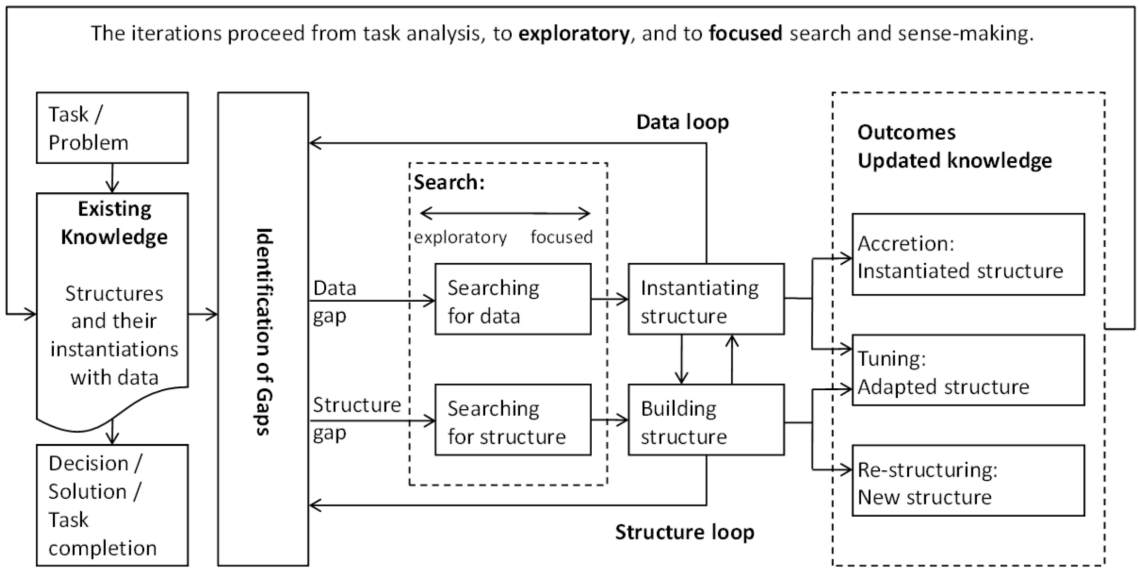


Figure 2.2: Zhang's comprehensive model of sensemaking

Inductive (data-driven, bottom-up) mechanisms	**Structure-driven (logic-driven, top-down) mechanisms**

• *Key item extraction*
Identifying key words/concepts (Kavale, 1980).
• *Comparison*
Comparing a concept to other concepts (Kavale, 1980).
  • *Similarity*
  Recognizing common features or attributes shared by concepts (Vosniadou & Ortony, 1989).
  • *Differentiation or discrimination*
  Recognizing different features of concepts (Chi, 1992; Vosniadou & Brewer, 1987).
• *Analogy and metaphor*
Analogical reasoning: concepts that share common features or belong to common categories may exhibit other common characteristics (Toulmin et al., 1979; Vosniadou & Ortony, 1989).
• *Classification*
Relating a concept to a broader conceptual category (Kavale, 1980) and grouping of sufficiently alike concepts.
• *Schema induction*
Discovering regularities in the co-occurrence of certain phenomena (Rumelhart & Norman, 1981; Vosniadou & Brewer, 1987).
• *Generalization*
Making claims about groups based on a sufficiently representative sample (Chi, 1992; Toulmin et al., 1979).
	• *Definition*
	Defining different aspects of a concept, such as purpose, function, and use (Kavale, 1980).
	• *Specification*
	Specifying conditions or requirements of a problem or task (Vosniadou & Brewer, 1987).
	• *Explanation-based mechanisms*
	Reasoning from cause: examining the causal connections of two phenomena (Toulmin et al., 1979).
	• *Elimination*
	Eliminating structures or facts that do not meet certain criteria in certain attributes (Kavale, 1980).
	• *Semantic fit*
	Examining the reasonableness with which a concept appears to fit a certain schema slot as it relates to the meaning of the knowledge structure as a whole (Kavale, 1980).
	• *Socratic dialogues*
	Critical dialogues to facilitate awareness of inconsistencies in the current schema. Recognition of anomalies can play an important role in initiating schema restructuring (Vosniadou & Brewer, 1987).
	• *Inference*
	Drawing a conclusion or making a logical judgment on the basis of circumstantial evidence and prior conclusions (Johnson-Laird, 1999).

Figure 2.3: Zhang's list of cognitive processes

- Problem-Centric Model: The schema organizing the evidence
- Unknowns / Residuals: The hypotheses with missing or unsupported information. This is closer to Russel's model [130]

Finally, the physical representations are digital artifacts utilized to support and manage this process on a computing device. Websites, forums, and search results are the online external sources of information that feed into the sensemaking process [7, 21, 153]. Tabs, clippings, and bookmarks serve as the evidentiary intermediary [29, 112, 135], and documents, spreadsheets, and diagrams are the tools through which users realize their models [79, 98]. Finally, ToDos, inline comments, and queued queries serve to represent current unknowns or residuals which have not been fully investigated yet [49, 146].

2.2 Sensemaking Systems

Over the past two decades, researches have developed a number of individual sensemaking support tools, designed around three different core areas: supporting document retrieval and filtering, enhanced reading and document triage, organization and structuring of content. Below I give a brief overview of a selection of these systems.

2.2.1 Search Support Tools

Some of the earliest tools designed to support online sensemaking revolve around assisting users with finding and managing information sources. These search support tools provide a wide range of support, from improving users ability to find the document they're looking for, to managing and revisiting their collected information sources. One of the earliest tools Scatter/Gather [39] utilizes document clustering to help users refine their document collection. Through a cycle of clustering and selection, users can achieve the right level of granularity in the documents they need to answer a particular question. Apolo [32] uses a similar technique with belief propagation, where users select individual documents of interest, instead of clusters, to drive further exploratory search in the domain. Intentstreams [9] utilizes an evolving set of keywords, rather than documents, to build out a stream of results that match more specific or tangential queries an individual might want to explore. Faceted Search [90] provides filters to end-users based on common, intrinsic properties of search results. These filters can also provide exploratory searchers with a broad understanding of some of the important features and dimensions in a particular information space. Lastly, DataShift [118] utilizes crowdworkers to augment the search process for queries involving non-traditional search media (such as images) and vague / unusual queries.

Two other tools, SearchBar [111] and Sensemaker [13] enhance the revisitation and refinidng experience. Sensemaker introduces collections of search results from one or more sources. End users can continue to build out or further constrain these collection by issuing additional queries. SearchBar, on the other hand, persists user's queries and results over multiple sessions. When a user resumes a sensemaking task after an extended period, they can use these persisted queries and results to resume their sensemaking activities.

2.2.2 Note Taking and Triage

Once users have a set of information sources they are working with, they then proceed to the process of triage or "active reading" [112]. During this process, users filter out irrelevant documents, read, and then markup and consume relevant documents as they build out their understanding of the space. TRIST [70] focuses on document-level triage, using techniques such as clustering, trend-analysis, and entity linking so a sensemaker can quickly focus in on relevant items. Other tools, such as VarifocalRead [86] provide an enhanced reading experience for large documents through three different zoom-level views. Lastly, InkSeine [62] and LiquidText [136] improve the document annotation experience through flexible markup, extraction, and summarization.

2.2.3 Organization and Structuring

Lastly, users often need to reconstruct the information they've found in useful format for display, consumption and sharing. Tools, such as the Visual Knowldge Builder (VBK) [131] and IdeaMache [98] utilize a free-form canvas (similar to a desktop) where users can position either whole portions or sections of their documents. They can then attach category groupings / labels onto those documents, giving them the capability to visual structure their information. Other tools further extend this desktop metaphor by adding features such as piles [102] or performing automatic topical clustering [5]. Lastly Hearst et al.'s tool [59] further enforces the cluster-based paradigm by having user assign one or more topics to a particular document. These can then be viewed in a "group view" as well as a "table view".

Chapter 3

The Knowledge Accelerator: Distributing Sensemaking

In order to better understand the global constraints surrounding the sensemaking process, we attempted to break apart and distribute individual portions of the process to crowdworkers in the prototype Knowledge Accelerator (KA) system. By imposing on ourselves the hard constraint of allowing contributions to only be performed through microtasks, we hoped to understand which key variables of the sensemaking process truly required a global overview and understanding. Then we wanted to explore which computational and workflow techniques could be used to manage that global context among the crowdworkers. Through the development of the prototype system, we were able to refine several different design patterns that can help to support not only crowdwork, but also individuals users with managing the context of a large task, such as sensemaking. We then tested the output of KA against the top Google search results for 11 different topics, and in aggregate found that the KA system was able to provide better answers across the dimensions of comprehensiveness, confidence, helpfulness, trustworthiness, understandability, and writing.

3.1 The Trouble with Microtasks

Microtasks offer an interesting alternative to conventional tasking, providing a way for workers to complete usable work in context free, bite-sized pieces. Because microtasks are quick to perform, they allow people to work without having to set aside large blocks of time and while mobile [14, 67, 115, 145]. Additionally, due to their limited context, they are easy to share with others and thus commonly used within the context of crowdsourcing [16, 34, 36, 91]. By decomposing and distributing the cognitive work of an individual, crowdsourcing can provide a larger pool of resources more quickly and with lower transaction costs than through traditional work.

However, much work in the real world is not amenable to crowdsourcing because of the difficulty in decomposing tasks into small, independent units. As noted by many researchers [16, 81, 99, 100], decomposing tasks – ranging from writing an article to creating an animated film – often results in pieces that have complex dependencies on each other. Take for example the goal of writing an article that synthesizes information

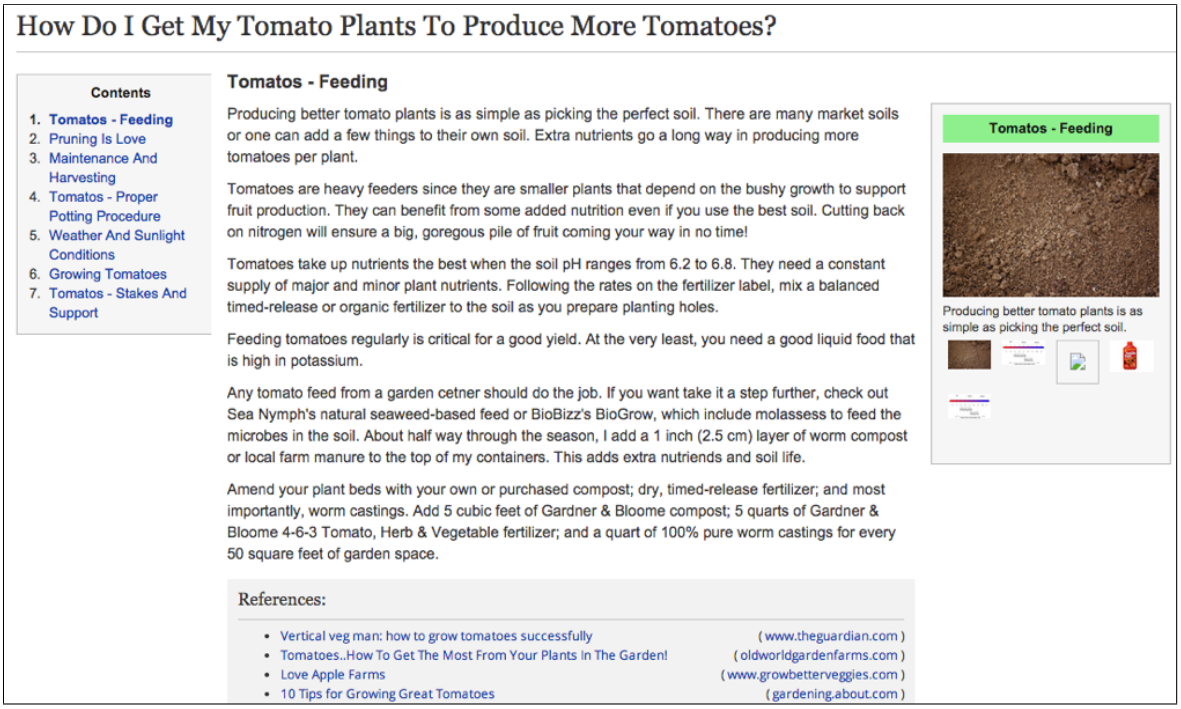


Figure 3.1: The final output of the Knowledge Accelerator system.

on the web about a given topic (e.g., growing better tomatoes). Coming up with a coherent and comprehensive set of topics (e.g., soil, sunlight, watering, pruning) is challenging without a global view of the data. The need for coherence extends throughout the fractal nature of the article: each section, paragraph, and sentence must have a proper transition and flow. Supporting such work requires having a big picture view of different pieces at different scales and ensuring they all fit together.

Accomplishing big picture thinking through microtasks is challenging because it means that each person can only have a limited view of the bigger picture. As a result, many of the applications of crowdsourcing have been limited to simple tasks such as image labeling where each piece can be decomposed and processed independently. Those approaches that do crowdsource tasks requiring big picture thinking — such as volunteer communities such as Wikipedia, open source software, or paid crowd work approaches such as flash teams [126] or Turkomatic [91] — have relied on a heavily invested contributor such as a moderator or an experienced contributor to maintain the big picture. For example, in Wikipedia a large proportion of the work is done by a small group of heavily invested editors [82], and the quality of an article is critically dependent on there being a small number of core editors who create and maintain a big picture structure for more peripheral members to contribute effectively [77].

In this chapter, we explore how a computational system, the Knowledge Accelerator, can scaffold an emerging interdependent, big picture view entirely through small contributions of individuals, each of whom sees only a part of the whole. Through a development of a working software system and an evaluation across a variety of topics, we were able to create a set of design patterns which can aid in the development of future systems dealing with the issue of a large global context.

3.2 Related Work

In the development of the KA system, we drew heavily from previous sensemaking models in the development of the distributed workflow. [40, 42, 53, 83, 109, 123, 130, 150]. Generally, the models agree that sensemaking is a dynamic and iterative process involving searching for information; filtering that information based on a user's goals and context; inducing a schema or structure from the information; and applying the schema to take action (e.g., writing a report, making a presentation).

A number of systems have been developed aimed at supporting these stages of sensemaking for an individual user [13, 43, 44, 95, 104, 118] or a group of users working together [77, 82, 114, 119, 120, 150]. However, prior research has focused almost exclusively on situations of integrated sensemaking in which individuals (even in groups) are heavily engaged in the entire sensemaking process. Instead, we sought to distribute the information synthesis process across many different individuals, each of whom may see only a limited view of the process.

3.2.1 Crowdwork: Complex Cognition and Workflow

While most crowdsourcing approaches have focused on simple and/or independent tasks, there is a growing interest in crowdsourcing tasks that tap into complex and higher-order cognition [78]. Many of these fall into the class of decomposing cognitive processing in a structured way such that many workers can contribute [3, 16, 20, 74, 76, 81, 91, 93, 94, 99]. Our work builds on this foundation by incorporating adaptive crowd workflows (e.g., TurKit, JabberWocky, CrowdWeaver), crowd-driven task generation (e.g, CrowdForge, Turkomatic), combining the outputs from decomposed tasks to create a global understanding (e.g., Cascade, Crowd Synthesis) and multi-stage crowd quality control process in which crowds can both generate new versions of output as well as vote on it (e.g., CrowdForge, Soylent, TurKit). However, we go beyond previous work in aiming to support a coherent big picture view while avoiding individual bottlenecks. Doing this is significantly more challenging than the tasks decomposed in prior research, requiring a search for structure during the sampling process, a reliance on novices to function with more context than they enter the task with, and a tight interdependence between each subtask such that any failures could negatively impact the value of the entire artifact. Computational Information Synthesis

Finally, some purely computational approaches have been explored for supporting information synthesis. For example, Question Answering (QA) research addresses the methods and systems that automatically answering questions posted by human in natural language. The complex, interactive QA (ciQA) has been introduced at TREC 2006 and 2007 in addition to factoid and list QA [41]. However, automated QA approaches (and their crowd-based variants [18]) focuses on answering short, factual questions instead of the complex sensemaking processes we are interested in, where users build up rich mental landscapes of information. Another approach is multi-document summarization [15, 54, 103, 107], which aims to use computational techniques to extract of information from multiple texts written for the same topic using feature based [55], cluster based [68], graph based [50] and knowledge based methods [56]. However, such approaches have limitations in dealing with complex yet short and sparse data that

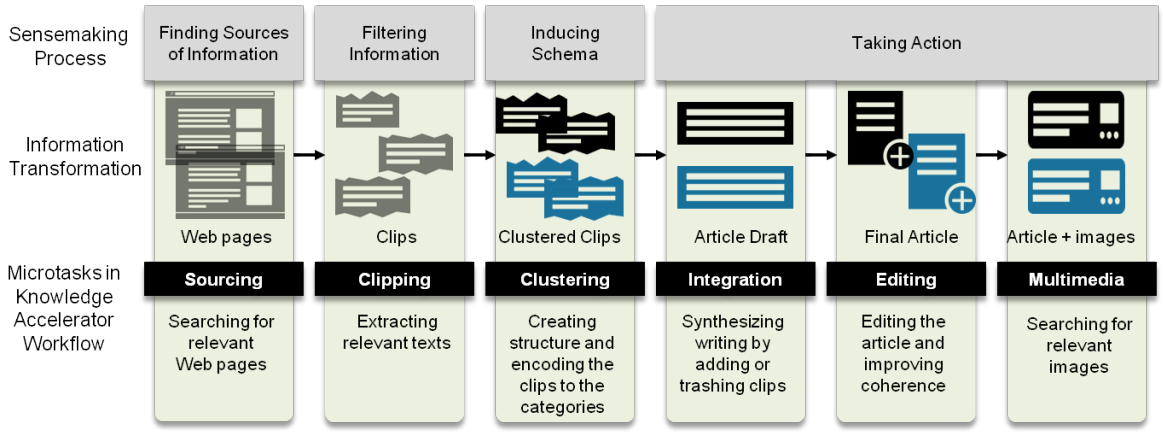


Figure 3.2: The process of the Knowledge Accelerator (KA), from start to finish

encountered on the web, and do not yet engage in the complex synthesis humans perform, which results in the cohesive and coherent output.

3.3 System Architecture

Broadly, there are two hard problems involved in crowdsourcing information synthesis: learning a good structure for the article based on sampling information from different online sources, and developing a coherent digest given that structure. In this section, we discuss how the Knowledge Accelerator system addresses each of these problems in turn.

3.3.1 Inducing Structure

How can a crowd learn a good structure for an article on an arbitrary topic? Previous crowd approaches such as CrowdForge or CrowdWeaver [76, 81] required workers to decide on a structure up before collecting information on each of these topics. However, these approaches fail when the structure must be learned from the data. For example, few workers will know what the subtopics should be for fixing a Playstation's blinking light or for dealing with arthritis; instead, the appropriate structure should emerge from the data. A single individual making sense of a topic often engages in an iterative process of sampling data and building a structure; however, to reduce the latency of having multiple cycles we explore an alternate approach in which the crowd samples a large amount of data in parallel, then leverage a novel hybrid crowd-machine approach that clusters information into topics without requiring any one worker to see the whole picture.

Finding Sources

To search for and filter high quality information sources we asked five workers to each provide the top five web pages relevant to the target question. We found these numbers to work well in practice; future work using optimization approaches [71] could

potentially set these dynamically. To ensure high quality responses, for each source we asked workers to report the search term they used and provide a small text clip as "evidence" showing why the source is helpful. This approach appeared to be successful in encouraging workers to find high quality sources: workers made on average 2 different queries ($\sigma = 0.3$), and their more commonly cited sources covered more categories of the structure with fewer sources than choosing sources using standard information retrieval approaches (i.e., using the MMR diversity-based re-ranking algorithm to reorder the sources gathered from the crowdworkers [23]). Sources cited by at least two workers were sent to the filtering stage.

Filtering Information

To filter relevant information snippets from each source, workers were presented with one web page and asked to highlight and save at least five pieces of information that would be helpful for answering the question using an interface similar to that described in [80] (Figure 3.3). One challenge we encountered was that each page could contain a variable amount of useful information, with some long pages having more snippets than a single worker would extract. To spread out worker coverage on long pages, we showed workers sections that had been highlighted by previous workers and asked them to first look for unhighlighted areas when choosing clips. This preference for novelty and surfacing prior workers' effort allowed us to engage multiple workers for tasks with an unknown amount of relevant information in a more efficient way than simply letting loose many independent workers who would overly focus on the beginning of the page, or having some workers start at the beginning and others at the end [16]. To focus more effort on potentially rich sources the system dispatches two workers to each source with an additional two workers for every two additional citations a source received.

Initially we had workers provide labels to categorize each clip, which we planned to use to develop a structure for the article. However, the lack of context of the bigger picture made these labels poorly suited for inducing a good structure. For example, in Figure 3.4 the top box shows the category structure induced from labels generated during clipping, while the middle and bottom boxes show the structure induced from the subsequent clustering phase and from a gold standard developed by two independent annotators with access to all clips and sources, respectively. Categories induced from the clipping labels poorly match the gold standard, and include categories with very different abstraction levels (e.g., *Use Drano Max Gel* vs *tips*). This motivated the development of the subsequent clustering phase.

Clustering

Inducing categories in unstructured collections of text typically requires understanding the global context in order to identify categories that are representative of the information distribution and at appropriate levels of abstraction. The problem of inducing structure without any single worker having a full global context is a particularly challenging problem, and although we describe a basic solution to the problem here for reasons of space and scope, we present a more sophisticated distributed approach in [31] that further generalizes the problem to other domains.

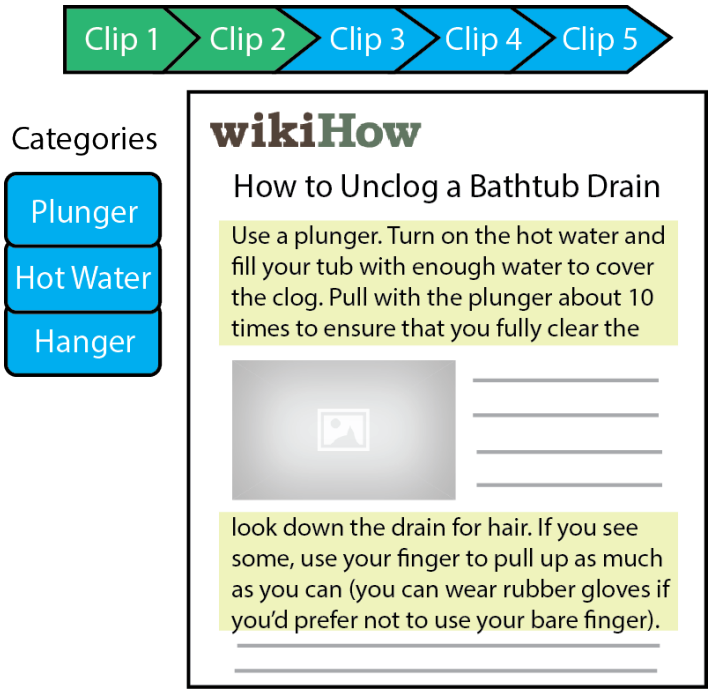


Figure 3.3: Workers extract 5 different pieces of relevant information from pages and give it a label

```
categories induced during clipping:
Boil Water, use hot water, Plunger, try a snake, How to Remove drain stopper,
bleach, Use Drano Max Gel, baking soda, drain, tips to unclog, problem, tools,
research, internet research, ..., etc.

---

categories induced after clipping:
Hot Water, Plunge, Plunger, Snake the Drain, Remove
the Drain Cover, Drain Cleaner, Remove Hair Clusters.

---

annotator categories:
Hot Water, Plunger, Plumbing Snake, Remove Cover, Chemicals, Bent Wire Hanger,
Call a Plumber, Shop Vacuum.
```

Figure 3.4: Categories induced from different stages for Q1: *How do I unclog my bathtub drain?*

Our approach takes advantage of the fact that many real world datasets have long-tailed distributions, where a few categories make up the bulk of the head of the distribution and many categories with few instances make up the tail. The intuition behind our approach is that first, the crowd can act as a guide to identify the large categories in the head of the distribution, with their judgments training a classifier to categorize the easy cases with high confidence. After automated classification, the crowd can again be used for "clean up", covering the low-confidence edge cases in the tail of the distribution. This also has the added benefit of easily breaking up the larger question context into sub-contexts for easier consumption in the later parts of the

system.

In the first phase, we use workers to label a number of representative categories and leverage those labels to identify meaningful features for an automated classifier. One critical challenge is that workers need to obtain a sense of the distribution of the data without seeing it all. To accomplish this we developed a design we call open-ended set sampling in which workers are presented with four random clips as seeds, and are asked to replace them repeatedly with another random clip until they can determine that the four seed clips belong to meaningfully different categories. Therefore, not only do they have to read the information present in the initial seed clips, but they also need to sample multiple times to understand what "different topics" mean for this dataset. In doing so they are randomly shown new clips, which means they are more likely to encounter categories with probability matching the distribution of topics in the data (i.e., higher probability of encountering larger categories).

After workers pick the seeds, we ask them to highlight discriminative keywords in each of the seed clips which are used to query for similar clips from the full dataset, which the workers then label as as *similar* or *different*. With the keyword highlights and the labels created by the workers, we use an SVM classifier and hierarchical clustering to cluster the high confidence portion of the dataset, sending the uncertain instances to Phase 2.

In the second phase, we employ crowdworkers to clean up the output of the classifier, by presenting them the existing clusters on the left of the screen, and the remaining clips on the right. The workers are first familiarized with the clusters by asking them to review the clips in each cluster and give it a short description. They then categorize the remaining clips into existing clusters or create new clusters if no existing cluster is relevant. These categorization judgments are used to refine the hierarchical clustering model.

3.3.2 Developing a Coherent Article

In this section we describe a set of processes which take as input a set of topics and clips for each topic and output a coherent Wikipedia-like article. There are two core challenges in doing this: first, creating coherence within a topic (e.g., consolidating redundant information); and second, creating coherence between topics (e.g., maintaining consistency across sections).

Integration

Within a single topic, there may be many clips which all contain substantively identical information (e.g., the ideal pH level of soil for growing tomatoes); one goal is to reduce this redundancy so that the final article only describes this information once. At the same time, we recognize the value to seeing that multiple sources all say the same thing; thus, we would like to keep track of all the sources that mention a particular chunk of information. Furthermore, tracking source provenance allows the user to drill back to the original information source in case it is described inaccurately or in a biased way.

To accomplish this we developed an interface in which workers were presented with 5 random clips of information for a given subtopic and asked to integrate that information

into a shared text pad. Specifically, they were asked to write the gist of the clip in their own words and transfer the provenance of the clip as a footnote. Missing footnotes triggered a verification check.

Initially, we just instructed individuals to cluster similar items together and insert only the footnote for redundant information. However, we noticed that workers were reluctant to change what they perceived as another worker's contributions, consistent with the social blocking found in Andre et al. [11]. This developed into a larger challenge: How could we get workers to gain an understanding of what was in the existing shared pad and feel comfortable modifying it? We introduced a technique we call evaluate then act that requires individuals to read what others have already put into the integrated answer before they are allowed to make a decision about the clip. Our final interface prompts workers to provide specific line numbers corresponding to existing information relevant to their clip, or to explicitly mark their clip as new information or trash. Compared to a version of the system without this structure, significantly more clips were inserted into the middle of the pad to align better to their given section (13% more, $t(24) = 2.568$, $p < 0.05$) or excluded (11% more, $t(24) = 4.592$, $p < 0.01$) when workers were asked to evaluate before acting.

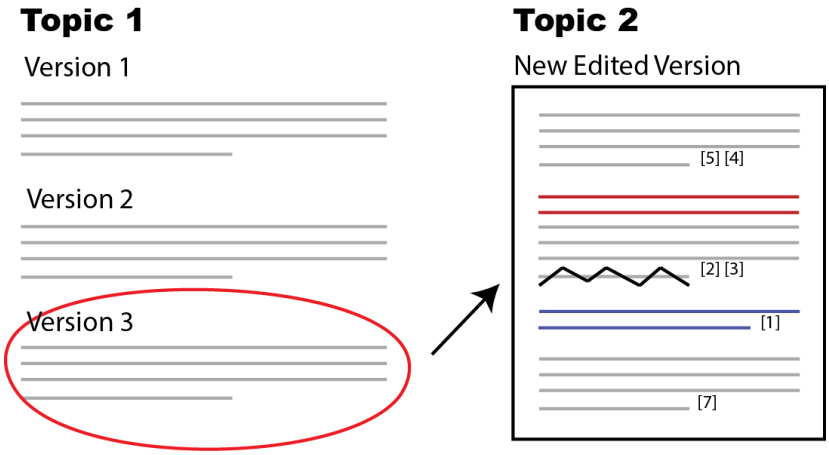


Figure 3.5: Editing users the 'vote-then-edit' pattern to promote consistency and motivate workers

Editing

We also noticed that coherence needed to be managed not only within topics, but between topics as well. A number of between topic inconsistencies became apparent during the development process, ranging from formatting to structuring to prose. For example, some topics would be organized with bullet points versus paragraphs, and some in the second person point of view versus third person. Previous crowdsourcing approaches have trouble dealing with cross-topic consistency because reading even a single topic can take significant time, let alone reading and editing across all topics. For example, CrowdForge's [81] approach simply concatenates topics into an article without any attempt at maintaining global coherence. This approach can succeed if the topics and structure either do not require consistency or if they are extremely well specified

beforehand: in CrowdForge and CrowdWeaver defining a science article "template" with clear sections such as *what is the problem, what the researchers did*, accomplishes this effectively in a similar manner to core editors specifying a structure in Wikipedia that peripheral members then fill in [77]. However, in the general case such well-defined and pre-specified templates are not always available.

To address this we introduced a new pattern which we call vote-then-edit (Figure 3.5). This pattern asks workers to first review and vote on and choose the "best" version of a subtopic created by previous workers, while simultaneously getting a sense for commonalities in style, grammatical choices, and organization. They proceed to edit a new subtopic (phase one) or improve on the item they voted on (phase two). In the second case, we expected workers would more carefully select the best version to reduce their future workload, as well as be more motivated to fix issues in it because they had a choice in what they wanted to do.

We used the vote-then-edit pattern in an interleaved "horizontal" and "vertical" workflow. The horizontal phase uses the refined and edited versions of a subtopic section as a "model" for improving the rough output from the integration phase for another subtopic section. Specifically, three workers vote on which of three versions of an edited subtopic section is the best and then edit a different subtopic subsection using their answer from voting as a model. Their resulting edited output is sent to the vertical phase, in which three workers vote on which of those versions is the best, and are then asked to further improve this now with all of the other subtopic paragraphs presented to them, to ensure the current subtopic has good flow with the other sections. The output from these workers is used in a new horizontal phase, and the cycle continues. The intuition here is that the horizontal phase provides only a single section as a model since there is substantive editing work remaining that requires relatively limited context, while the vertical phase provides all sections because the primary editing work remaining is ensuring consistency across sections. Splitting editing into two interleaved phases with different context-work tradeoffs appeared to be more effective than an older editing approach with a single phase. When we compared the evaluation ratings for the older editing to the interleaved vote-then-edit approach for two questions (Q1 and Q2 in Table 3.1 respectively), the newer answers were found to be significantly more understandable ($\bar{x} = 0.457$, $p < 0.01$) and helpful ($\bar{x} = 0.373$, $p < 0.05$), suggesting this design pattern helped to create more coherent output.

Multimedia

Images and video can help the reader skim and digest information quickly, as well as provide rich information such as diagrams, instructions, and how-to examples. In our system we enable multimedia from diverse sources to be tied to information blocks, which we define as sections of text demarcated by footnotes. Informally, information blocks correspond to units of information, such as steps in a how-to, or statements or evidence. This has the benefit of ensuring that the images found are specific to pieces of information found in the answer, rather than just being general to the subtopic. For the version of KA described here we did not employ redundancy or voting in the multimedia stage as we did not encounter quality issues; however, since multimedia enrichment is not a particularly interdependent task existing known quality control approaches such

as redundancy and voting [78] would likely be sufficient for a production system.

3.4 Design Patterns

As mentioned in the above task descriptions, during our iterations on each stage we ended up introducing several design patterns that improved the output. Each phase had its own distinctive challenges, yet they still suffered from some of the core challenges highlighted by previous work: motivation, quality-control, and context [78]. Our design patterns served to guide our final system design and add to the set of crowd patterns introduced by previous research [16, 20, 78, 81, 91, 92, 99]. They may be particularly relevant for challenges involving complex interdependent tasks requiring global context for workers seeing only local views.

3.4.1 Context before Action

One of the biggest challenges in crowdsourcing a complex, interdependent task such as information synthesis is providing workers with sufficient global context to perform well despite them having only a local view. Previous researchers have suggested a variety of useful patterns related to this goal, including making the cost of spurious answers as high as valid ones [75], identifying and surfacing specific sub-task dependencies [91, 126], unified worker interfaces [158] and re-representing tasks in simplified forms [10, 76]. We contribute a set of patterns adding to this literature, specifically focusing on a key tradeoff: given a limited amount of time and effort for an individual worker, how can we provide workers with global context (i.e., investing in their ability to make better decisions) but also engage them in actual production work? Too much invested time providing context reduces the amount of time available for improved task performance.

Open-ended Set Sampling. One challenge with large datasets is giving workers a sense of the distribution of the data despite their observing only subsets of it. This pattern involves a comparison task in which workers are asked to sample random items from the data in order to create a set of non-matching items, as seen in the first step of clustering. A key design factor in this pattern is having a good set function that provides a driver for open-ended sampling and also a stopping point (e.g., when a worker's familiarity with the distribution gives them a sense that their four seeds represent substantively different topics in the dataset).

Evaluate then Act. In order to get workers to understand the context provided to them, we designed evaluation mechanisms at the beginning of their main task that would allow them to get acquainted with the output from previous workers. This helped workers understand how previous workers processed the information provided to them, improving consistency of the output on parallel tasks, and reducing repeated information. This pattern was leveraged in a number of tasks: clustering, integration, and editing. In the integration phase, we additionally used the evaluation phase to signal to workers that removing others' work was acceptable and expected, showing that it could be useful in socializing workers into desired procedural practices as well as providing them with context.

3.4.2 Tasks of Least Resistance: Leveraging Worker Choice

Since workers were mostly dealing with dense textual information on a topic they were likely unfamiliar with, we wanted to ensure they were sufficiently motivated. Therefore, we developed a pattern that doubled as both a quality control measure, we well as an incentive for workers. The "task of least resistance" pattern requires that the same crowd worker be involved in two stages of the task, a first stage in which they choose what to work on from a number of alternatives (e.g voting) and a second stage in which they themselves benefit from their choice in terms of having to do less work, easier work, or being able to submit a higher quality output. The intuition is that to minimize their later work workers will choose a foundation that requires the least amount of work possible; i.e., they will choose the "task of least resistance". This act of choosing is intended to also provide workers with a sense of agency and purpose, which has been shown to increase task performance [26, 128]. This choice also has the potential to increase task performance through workers trying to avoid cognitive dissonance: since workers have themselves presumably chosen the best quality work to start, poor quality final output could reflect on their own worth [149]. This has a trade off of potentially making tasks longer, more complicated, and more expensive, however the benefit is a higher quality output.

3.5 Implementation

The main portion of the application was built using Ruby on Rails and integrated with Amazon's Mechanical Turk through the Turkee ruby gem [69]. The Ruby on Rails application served as the primary user interface for both the question asker, crowd worker, as well as the answer viewer. A question posed to the system would start the workflow, beginning with source finding. For each stage, after a certain set of conditions were met (number of sources, clips, completed clustering, etc.), the next task in the workflow was automatically started. This allowed the system to run through the entire process with minimal intervention.

The clipping task utilized Readability's parser API to simplify the appearance of the sources provided during the sourcing phase. This allowed workers to view a cleaner interface in which to clip from, and it also removed some technical limitations involved with clipping from pages that might be multi-paged (readability combines these into one long document) or featured heavy javascript functionality that would interfere with the clipper tool.

For the first phase of the structure induction tasks, the TfIdfSimilarity ruby gem is used for searching clips similar to the seed clips [108]. LIBSVM is used for combining the crowd judgments and cluster a large portion of the dataset [27]. For the integration and editing tasks, we utilized the Etherpad-lite text pad library [141] to allow workers to simultaneously work on the same output.

3.6 Evaluation

To evaluate the usefulness and coherence of the system's output we compared it to sources an individual might use if they were to complete this task without the KA system. This would most likely involve the use a search engine such as Google to gather information and use existing information sources to learn about the topic. Therefore, as an evaluation, we had a separate set of crowd workers perform a pairwise comparison of the KA output to that of top results returned by Google and those found useful by multiple crowd workers.

3.6.1 Method

Participants were recruited through the AMT US-only pool and paid $1.50 for the evaluation task. Each participant was randomly assigned to compare the output from the KA system with an existing top website for that question. An individual could only provide one rating per question, but could do the rating task for more than one question. We removed 34 of the 1385 unique participants who provided an evaluation rating who also participated in a KA system task.

The "top websites" used in the comparison task were the top five Google results, as well as any additional Google results that were highly cited (mentioned by 3 or more turkers) during the sourcing phase of the system. Some questions had a larger number of highly cited sources, resulting in more additional websites, as can be seen in Figure 3.6.

In the evaluation task, participants were first asked a series of questions that would cause them to read and understand both sources. In order to encourage quality through defensive task design [75], for the output from the KA system and the existing web page, they were asked to list the different sections on each and three different keywords that would describe those sections. After they read and parsed each web page, they were presented with a brief persona of a friend who was having the problem posed to the KA system. Workers were then asked, for that problem, to rate the comprehensiveness, confidence, helpfulness, trustworthiness, understandability, and writing of each web page on a seven point Likert scale (from 1 to 7) and provide an explanation for their rating on each dimension. We averaged ratings on these dimensions into a single score representing the overall perceived quality of the page.

We selected 11 target questions for evaluation by browsing question and answer forums, Reddit.com, and referencing online browsing habits [25]. For some questions, we added some additional constraints to test the performance of the system for more personalized questions. In addition to this external evaluation, we also had the crowd-workers who participated in the KA system fill out a short feedback form detailing their experience using the system. We ask three questions about the difficulty of the task, the clarity of the instructions provided, and the easy of use of the user interface. We recorded some brief demographics about our workers, including to the country they were from.

Question	N	Score
Q1: *How do I unclog my bathtub drain?*	116	0.292 *
Q2: *How do I get my tomato plants to produce more tomatoes?*	177	0.420 *
Q3: *What are the best attractions in LA if I have two little kids?*	158	-0.044
Q4: *What are the best day trips possible from Barcelona, Spain?*	98	-0.109
Q5: *My Worcester CDi Boiler pressure is low. How can I fix it?*	139	0.878 *
Q6: *2003 Dodge Durango has an OBD-II error code of P440. How do I fix it?*	138	0.662 *
Q7: *2005 Chevy Silverado has an OBD-II error code of C0327. How do I fix it?*	135	0.412 *
Q8: *How do I deal with the arthritis in my knee as a 28 year old?*	139	0.391 *
Q9: *My Playstation 3 has a solid yellow light, how do I fix it?*	119	0.380 *
Q10: *What are the key arguments for and against Global Warming?*	138	0.386 *
Q11: *How do I use the VIM text editor?*	138	0.180
* = significant at $p < 0.01$ after Bonferroni correction		

Table 3.1: Average difference between the KA output and top websites for the eleven questions (positive indicates higher ratings for KA, negative indicates higher ratings for the competing website). Each rating was an aggregate of 6 questions on a 7-point Likert scale.

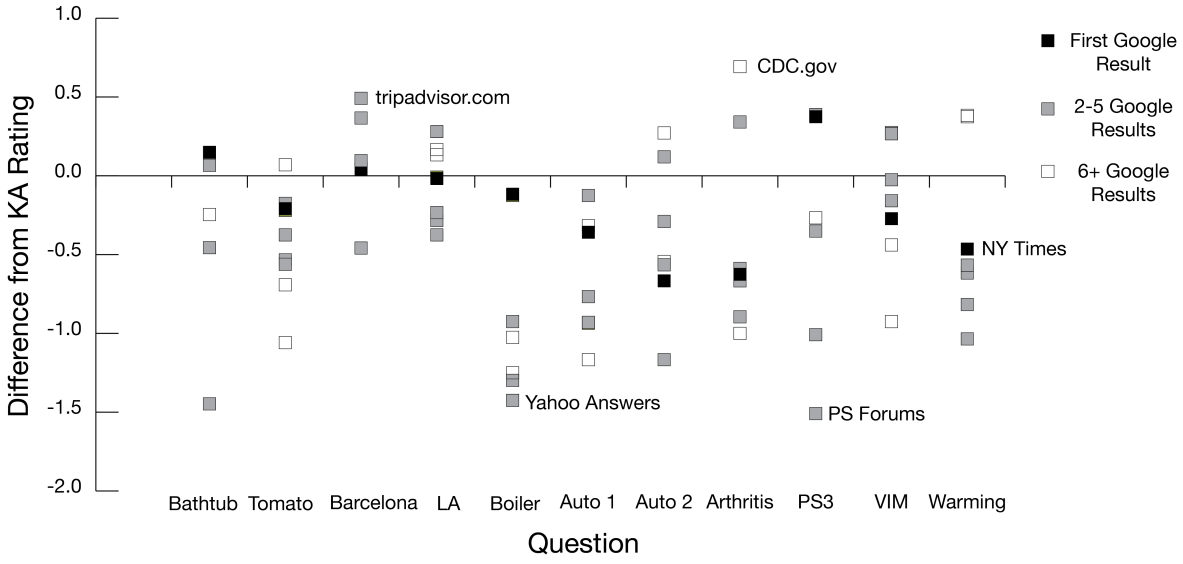


Figure 3.6: Results across questions and websites. Points represent the average aggregate score difference between the KA answer and an existing site

3.6.2 Results

Aggregating across all questions, KA output was rated significantly higher than the comparison web pages, which included the top 5 Google results and sources cited more than 3 times (KA: $\bar{x} = 2.904$ vs Alt. Sites: $\bar{x} = 2.545$, $t(1493) = 13.062$, $p < 0.001$). An analysis of individual questions corrected for multiple comparisons is shown in Table 3.1.

The strongly positive results found were surprising because some of the websites in the comparison set were written by experts and had well-established reputations. Only on the two travel questions, Barcelona ($\bar{x} = -0.109$) and LA ($\bar{x} = -0.044$), and the VIM question ($\bar{x} = 0.180$) did the KA output not significantly outperform the comparison pages. A closer examination of these pages suggests that for the two travel questions, because of the strong internet commodity market surrounding travel, a considerable amount of effort has been spent on curating good travel resources. Even with the slightly more specific LA query, there were still two specialized sites dedicated to attraction for kids in LA (Mommypoppins.com and ScaryMommy.com). The VIM question represented a mismatch between our output and the question style. A number of the sources for the question were tutorials, however in the clipping phase, these ordered tutorials were broken up into unordered clips, creating an information model breakdown. This points out an interesting limitation in the KA approach, and suggests that adding support for more structured answers (e.g., including sequential steps) could be valuable future work.

As an additional external evaluation, for the two questions (Q6 and Q7) related to automotive systems we compared the discovered categories from the KA system with two commercial knowledge service products generated by expert technicians. We compared the KA response's accuracy and comprehensiveness, and found that it discovered all the categories referred to in these two commercial products for each question. Furthermore, the categories from the KA output provided more categories not mentioned in the commercial product (average 2.5 categories from two commercial products, while average 9.5 categories from KA). We validated these additional categories with expert automotive professionals who evaluated them as also being plausible and reasonable for the given questions. There was one instance in which two distinct categories (Encoder Motor and Encoder Motor Sensor) from the commercial products were clustered into the single category named Encoder Motor Assembly in the KA output. However, the full text answer from the KA system for Encoder Motor Assembly did still contain these two sub-components with different repair procedures.

It may seem surprising that KA would work well for questions such as automotive error codes, where the response relies heavily on technical knowledge and jargon. On further inspection we believe this is because there are many online resources that have valuable information pertaining to these questions but are in unstructured and dialog oriented forms. Workers in the sourcing phase found rich sources of online information from many car enthusiast discussion forums, in which members tried to diagnose and help each other solve their automative problems. Although crowd workers may not understand the esoteric jargon of the automative domain, their understanding of grammar, semantics, and argument structure was sufficient to let them find, filter, cluster, integrate, and edit this domain-specific information. These results suggest a interesting avenue for future research leveraging human understanding of semantics and

argument structure to extend crowdsourcing to process expert domain knowledge and to understand the limits of where such an approach breaks down.

On average, running a question through the KA system cost a total of $108.50 (see Table 3.2). Although our primary goal was to establish a proof of concept of accomplish big picture thinking in small pieces, we return to the issue of cost in the Discussion. From the self-report crowdworker feedback, workers mostly found the tasks to be easy to complete, with the clustering phase having the most difficult task.

Phase	Task Pay	Avg. # of Tasks	Avg. Cost
Sourcing	$0.25	15	$3.75
Clipping	$0.50	21.6	$10.80
Clustering 1	$1.00	10	$10.00
Clustering 2	$1.00	10	$10.00
Integrate	$0.50	37.2	$18.60
Edit 1	$0.75	28.8	$21.60
Edit 2	$1.00	28.8	$28.80
Images	$0.50	9	$4.50
Total		160.4	$108.05

Table 3.2: Average number of worker tasks and average cost per phase, and overall, to run a question.

3.7 Discussion

The strong performance of the system is perhaps surprising given that its output was generated by many non-expert crowd workers, none of whom saw the big picture of the whole. We do not believe that this should be interpreted as a replacement for expert creation and curation of content. Instead, the power of the system may actually be attributable to the value created by those experts by generating content which the crowd workers could synthesize and structure into a coherent digest. This explanation suggests that the approach would be most valuable where experts generate a lot of valuable information that is unstructured and redundant, such as the automative questions in which advice from car enthusiasts was spread across many unstructured discussion forums. In contrast, KA's output did not outperform top web sources for topics such as travel, where there are heavy incentives for experts to generate well structured content. We believe its performance is likely due to its aggregation of multiple expert viewpoints rather than particularly excellent writing or structure per se, though this is a fruitful area for future investigation.

In developing the KA system, we explored a number of approaches that did not work. We initially tried to avoid a clustering phase altogether by exploring variations of the clipping task in which we provided additional context to workers in having them read through multiple sources, engage the workers who found sources in doing the clipping, or have them build on the categories that other workers had already generated rather than work independently. However, in all cases workers did not generate good labels due to a lack of context. We then explored introducing an additional "conductor"

view, in which workers could be recruited as clips came in to organize those clips and close categories that had a sufficient number of clips; however, this also failed because the conductors did not have sufficient global context to create good categories. These failures motivated the hybrid crowd-machine clustering phase.

Development of the integration and editing phases also included many false starts due to the opposite problem of giving workers *too much* context. Our first integration interface enabled multiple workers at the same time to easily view and expand all the clips in a category for within-category context, and also see the current state of how other categories were developing for between-category context. Our idea was that as workers integrated clips and built out more options exposure to the other clips and options in real time would help them create more coherent digests. However, this approach proved overwhelming for scaling up to a large number of crowd workers engaged for short time periods. This motivated us to split up within-category and across-category consistency into the integration and editing phases and the development of the vote-edit pattern.

We encountered a number of places where our approach could be improved. As evidenced in the VIM question, the lack of support for nuanced structure in our digests can prove problematic. For some sources such as tutorials or how-tos, supporting sequential dependencies between steps could be useful. While our output was able to support such dependencies in an ad-hoc way within a category (such as the sequential steps for plunging a drain) it would be profitable to be able to support sequential dependencies across categories (e.g., first try x, then try y). More structure could also be beneficial for particular domain areas, such as explicitly capturing symptoms and causes as different types for automotive or medical diagnostic questions.

The system could also benefit from including iteration. For example, after workers completed the integration phase they were asked the question "What else needs to be done to make this a complete answer?". While many obviously said the section needed be edited, one of the most popular responses was "Needs more information." This suggested to us that while our clips and categories had pulled in most of the information, there was more information in some sections we were missing. One possibility is to introduce an iterative component at this point – as workers are integrating information into the pad and notice missing information, they can request for other workers to go out and find that additional information through clipping. Thus while the system was partially successful at taking a breadth-oriented approach rather than the deeply iterative approach typical of sensemaking [42, 44, 123, 130], understanding how to best incorporate iteration would be a valuable area for future work.

Aside from improving the quality of the system output, there is also the possibility of reusing the output for other users researching similar questions. Although users have complex information seeking needs, many of the queries they issue are similar. For example, a recent study estimated that 3% of search queries account for 1/3 of total search volume [151]. Thus at a minimum, many answers could be amortized across users with the same question. A particularly promising but challenging opportunity is if similar questions may be able to reuse components of already summarized answers; for example, a question on investing advice for a 50 year old might use some common categories as for a 20 year old, but others would be unique to the new question's context. Challenges for the reuse of information are how the system would be able to identify the

similarity for possible answers during each information synthesis phase and what level of granularity should be considered to for an effective system. Spatial and temporal reasoning over the existing knowledge and new information could be considered to provide context-aware and up-to-date answers.

We hope the design choices embodied in the KA prototype system and the design patterns discussed here may be useful for other system designers working to distribute cognitive complex tasks. Some domains that might benefit from this include micro-task markets, which could benefit from supporting more complex tasks; volunteer crowdsourcing efforts such as Wikipedia [77] or friendsourcing in which many small contributions are readily available [17]; or self-sourcing in which the crowd within could accomplish complex tasks in small increments (e.g., waiting for the bus) without needing to load the entire task context into working memory [140]. Overall, we believe this approach represents a step towards a future of big thinking in small packages, in which complex and interdependent cognitive processes can be scaled beyond individual cognitive limitations by distributing them across many individuals.

Chapter 4

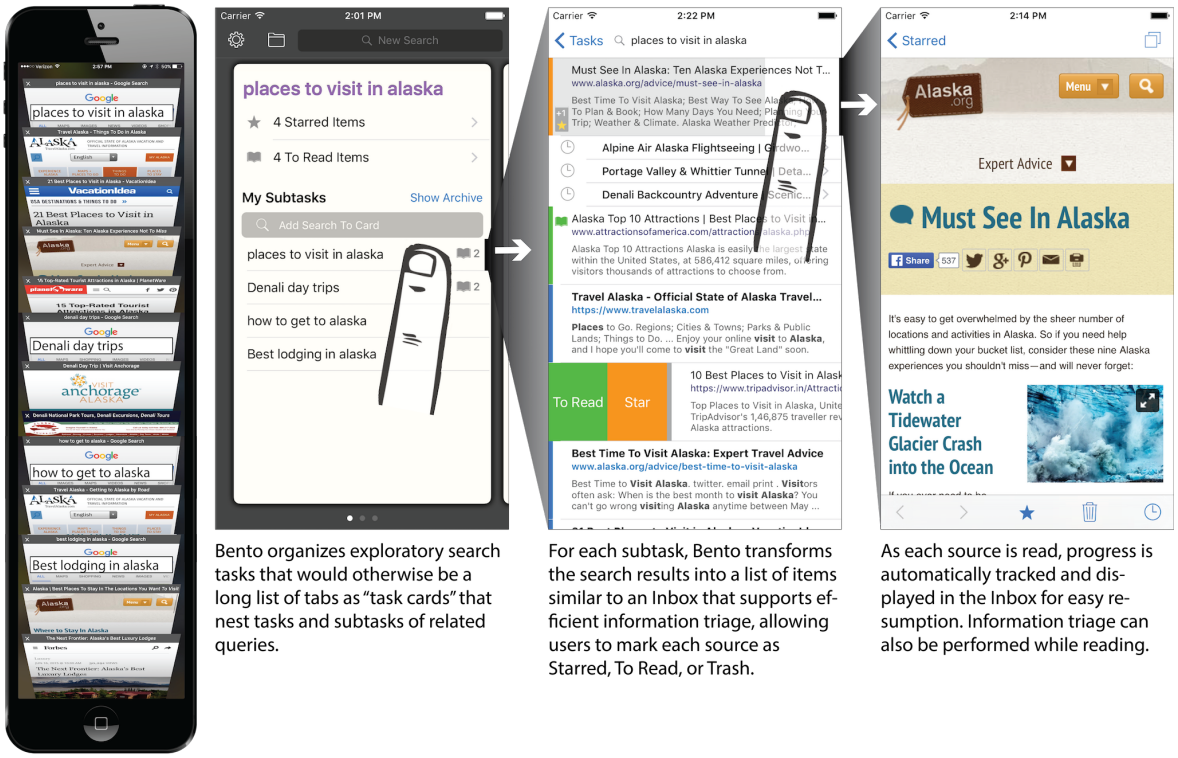
Bento: Source Management

Through the Knowledge Accelerator work, I was able to identify a reasonable workflow that was able to support distributed sensemaking across a group of crowdworkers. However, this workflow was able to work because all the individuals in the process were extrinsically motivated to do so. In an individual scenario, the overhead introduced by such workflow driven tools might not offer an easily perceivable intrinsic benefit over current methods. Rather, by focusing on supporting not just the workflow, but the set of cognitive mechanisms occurring from one transition in the workflow to the next, enough value and benefit can be provided to users to encourage tool adoption.

By using the KA workflow as a guidepost for breaking apart the different activities and phases of the sensemaking process, I began to consider ways to provide in-situ support for the sensemaking process,. In this next set of chapters, I discuss a few different systems and tools I built to help provide this support for certain portions of the sensemaking process: Bento for helping to manage sources, Siphon for streamlining source clipping, and Distil for assisting with structure generation. The first system I discuss, Bento, looks at the processes of search and triage, and introduces several mechanisms to support those activities over multiple sensemaking sessions.

4.1 Introduction

As users begin their information foraging process, they often look to multiple sources of information to provide a clear picture of what the entire information space looks like: which options they can choose from, the full scope of a topical area, and all the different opinions around a subject [90, 104]. Due to the plethora of information, users divide their attention among these sources by going through a triage process - they perform a lightweight evaluation of the information available, and if deemed potentially useful, will either mark it for follow up, or dive deeper into it [12, 106]. In modern browser environments, this behavior largely occurs through the tabbed interface: users will queue up and manage potential sources of information. Consider a person planning a trip to Alaska: on a desktop they may create multiple tabs for each location or point of interest, which quickly multiplies as the user drills down into restaurants, hotels, and activities for each of those locations – potentially resulting in dozens of tabs open at once. Adding complexity to the situation, many of these foraging tasks may be going



Bento organizes exploratory search tasks that would otherwise be a long list of tabs as "task cards" that nest tasks and subtasks of related queries.

For each subtask, Bento transforms the search results into a list of items similar to an Inbox that supports efficient information triage, allowing users to mark each source as Starred, To Read, or Trash.

As each source is read, progress is automatically tracked and displayed in the Inbox for easy resumption. Information triage can also be performed while reading.

Figure 4.1: Comparing a typical list of tabs (left) with Bento's search centered navigation from the same exploratory search task.

on in parallel (e.g., investigating alternate destinations such as Anchorage vs. Homer), may be suspended and resumed in various states of progress over time, and may be interleaved with other tasks (e.g., finding a place to eat tonight).

In this chapter I discuss Bento, and mobile interface for sources management. We performed this exploration using the practical and interesting design constraint of a mobile device. Mobile devices are used now more than ever for information seeking activity [45], however they are significantly smaller, are operated in short bursts of time, and activities are frequently interrupted on them [14]. Addressing sensemaking in a mobile device context thus is not only timely and important, but provides additional generative constraints for new approaches.

Through this work, we introduced an alternative approach to tabbed browsing for source management that also addresses the additional constraints involved in a mobile context. The key insight we built on is that tabs are often performing two distinct functions: 1) they serve as a way to organize and juggle multiple tasks that may be going on at once; and 2) they serve as a workspace to triage and build a mental model for a given task, for example queuing sources for later consumption, performing comparisons between sources and saving information of uncertain value for further review. Because tabs are overloaded in such a manner, we argue that they accomplish neither task very well, especially when used in a constrained mobile environment.

To overcome the limitations of tabs, we introduced a scaffolded process that separates the task management and workspace functions of tabs into two distinct interfaces. Instead of having many open tabs we transformed the search results page into a mutable

workspace that allows users to triage and keep track of their progress on any given search, with those searches collected into tasks and subtasks. We instantiated this approach in a novel mobile web browser, Bento Browser, and evaluated its effectiveness through three user studies. Our results suggest opportunities for the development of novel systems of online information seeking for both mobile and desktop platforms that both better suit the nature of complex searching as well as constrained mobile environments. Mobile Sensemaking First, we wanted to gain a clearer picture of how users were using their mobile devices for exploratory search. To explore this we performed a short survey, first partially published in [29], with 164 smartphone users (98 Male, 66 Female, Age: M= 32.29, SD= 8.72) on Amazon's Mechanical Turk platform. We asked a series of questions about a user's exploratory searches, how often they perform them, what were some past searches, as well as the interface tools they use. Surprisingly, we found that people reported frequently conducting complex exploratory searches either partly (70%) or completely (45%) on their phones, ranging from planning a vacation to researching woodworking projects. However, 47% of the users also agreed with the statement that "It would be frustrating to do a complex search on a smartphone".

We asked participants a number of questions about their current habits, based on a 5 point Likert scale (Rarely - A Great Deal). When queried about which exploratory search activities they currently perform on their phones, the most common activity was simply "Reading web pages" (M = 4.03, SD = 0.89). Text entry during search (M = 3.59, SD = 1.14) and keeping track of multiple pages (M = 3.24, SD = 1.12) were the next to most common activities, with saving web pages (M = 2.40, SD = 1.17) and collaborating (M = 2.21, SD = 1.14) being the two most uncommon activities.

We then asked about future support. 80% of participants agreed with the statement "I would find it valuable if smartphones had better interfaces for doing complex searches". Delving deeper into this question, at least 1/3 of the respondents reported extreme difficulty (highest Likert rating) with "saving web pages", "keeping track and switching between pages" and "sharing findings with others". These suggest that the current browser interfaces on smartphones do not well support the constant context switching and task suspension present in exploratory search. Conversely, participants cited the advantage of being able to do searches "on the go" and the general "convenience" of smartphones. These results suggest users think there are significant problems with managing exploratory searches on smartphones, even though they currently do them, and would like to continue to do them. This suggests addressing complex searching in the mobile context may have both real world practical value as well as being a source of potentially generative design constraints that could also translate to less constrained device footprints such as the desktop.

4.2 Understanding Tabs

Tabs are a ubiquitous feature in every major web browser today, where they serve multiple functions ranging from organization to triage to reminding [49, 65]. In particular, they serve two primary functions in the exploratory search process. First, tabs provide task management functions – by separating out tasks [148], acting as a reminder to resume a task, and allowing for quick efficient switching between tasks [65]. Second,

they provide a workspace for triaging sources [59], performing comparisons between sources [80], and saving good resources for further review.

We note three specific problems with the ways that tabs try to support these two function simultaneously. First, tabs are only loosely coupled to their generating activity. As a result, tabs during exploratory search become disconnected from their search results page, potentially causing negative effects such as users losing track of why they opened a tab, where they were in their task progression, and which pages belong to which tasks [49]. This is particularly problematic early in the exploratory search process as users are uncertain about the future value of the information contained within them [80]. Second, tabs are ordered based on the sequence in which they were opened and which tab spawned them, in order to keep them co-located to the other tabs in their task. This can become inconsistent as an organizational model as tabs are closed or opened in the middle of other tabs, and also misses an opportunity to provide more meaningful organizational structure, either for separating tasks or as a workspace. Lastly, tabs have limited context (e.g., a favicon and partial title) which can make it difficult to find a tab, know the state of progress on using it, or understand which tabs belong to which tasks. All three of these challenges are exacerbated in the mobile context, where there is little space to show multiple tabs at once or to provide context for them.

4.3 System Design

Seeing the issues surrounding tabs, we developed Bento, a novel interface for scaffolding complex search which obviates the use of tabs while still supporting their underlying task management and workspace functions of tabs.. This can be seen in see Figure 4.1, which shows how a user might perform planning a trip to Alaska with the current paradigm of tabbed browsing on a mobile device versus Bento. The fundamental component enabling the approach is transforming the search results page in place into a mutable workspace that allows the user to queue page to read (analogous to the common practice of opening a search result link in a new tab), star pages they found useful, trash unhelpful pages, and, critically, to see the progress they have already made in reading each page they opened (See Figure 4.1). Unlike previous approaches (e.g., collaborative search [113], history management [6, 111], or activity workspaces [60, 73, 147]) which require a separate interface for managing and surfacing individual tabs, Bento's approach provides a natural centralized workspace in the search results page that is already a fundamental and familiar element of navigating complex searches and obviates the need for tabs altogether.

Of course, for complex searches a single search is often not enough; for example for planning a trip to Alaska one might have additional searches for day trip destinations, how to get there, and where to stay. For managing tasks and subtasks Bento bundles search result pages together into task cards, drawing inspiration from previous search-based task management tools such as SearchBar [111]. However, one difference from tools which focus on surfacing the past history of searches is the prospective nature of Bento's scaffolding, in which users can (and did) create searches as placeholders and reminders of subtasks they would need to work on (like finding a place to stay) before actually doing any of the work. Together, these elements suggest a radically different

way for people to manage complex searching than traditional tabbed browsing. Below we describe the design rationale for developing Bento and details about its various interface elements.

4.3.1 A Sensemaking Workspace

When creating this workspace, we initially considered leveraging approaches utilized by previous information triage systems (e.g., [59, 131]). However, these tended to rely on spatial organization which were not a good fit for the limited real estate of smartphones. After a number of design iterations, we settled on a representation evoking the affordances of an email inbox. Email inboxes are designed for quick and efficient triage by users, providing information to users about what information is important, has been dealt with, and what still needs to be read / triaged. They accomplish this in a simple list format – not requiring the larger spatial requirements of other information triage systems. Email inboxes provide users with organization strategies ranging from flagging or starring items (which can pin them to the top of the list), archiving undesired items, and marking items to be read later (e.g., through marking as unread). We found these strategies useful for organizing searches, allowing search results to be flagged as important, archived if irrelevant or not needed in the future, marking items as potentially relevant and of interest to come back to, and supporting an awareness of where search results are in the list (in this case, ranked by relevance to the query if acted on, or in their original search result order if not). The items in the inbox are ordered with starred items at the top, followed by to read items, and finally any other search results in their original relevancy order. Trashed items are placed at the very bottom of the search results list to enable undo if needed. The search results also have a natural progression of states: unviewed search results show up with bold text and a blue dot similar to an “unread" email message. Users can then manually mark a page as being in an intermediate state, with a ‘to read later’ annotation, or as a particularly useful reference source, with a ‘star’.

We not only considered triage to be important in this interface, but also resumption and information provenance (See Figure 4.2). Resumption is managed through a couple of factors: the search results persisting in appearance, read / unread indicators, and progress bars. Initially, we combined the progress indicator and read/unread indicator into the same space, however this caused a number of misinterpretations or disregard of the progress indicator all together. To increase visibility, we separated the progress indicator from the priority indicator (see Figure 4.4). We instead represented the progress indicator as the background fill of each search result. As the user scrolled further down the page for a result and spent more time on that result, a gray bar would fill up the background of the row. In early piloting users found this to be much more intuitive and easily parsed. The read / unread indicator was adjusted to be a colored bar on the far left of the row. Additionally, when a user reopens a page, they are automatically scrolled to their last position on that page, letting them quickly resume what they were reading. We believe this novel approach of showing progression information directly on an information source gives users a way to understand progress without having to visit the source.

Lastly, to maintain information provenance, all subsequent pages visited from a

Figure 4.2: The different manipulations that can be applied to a search result

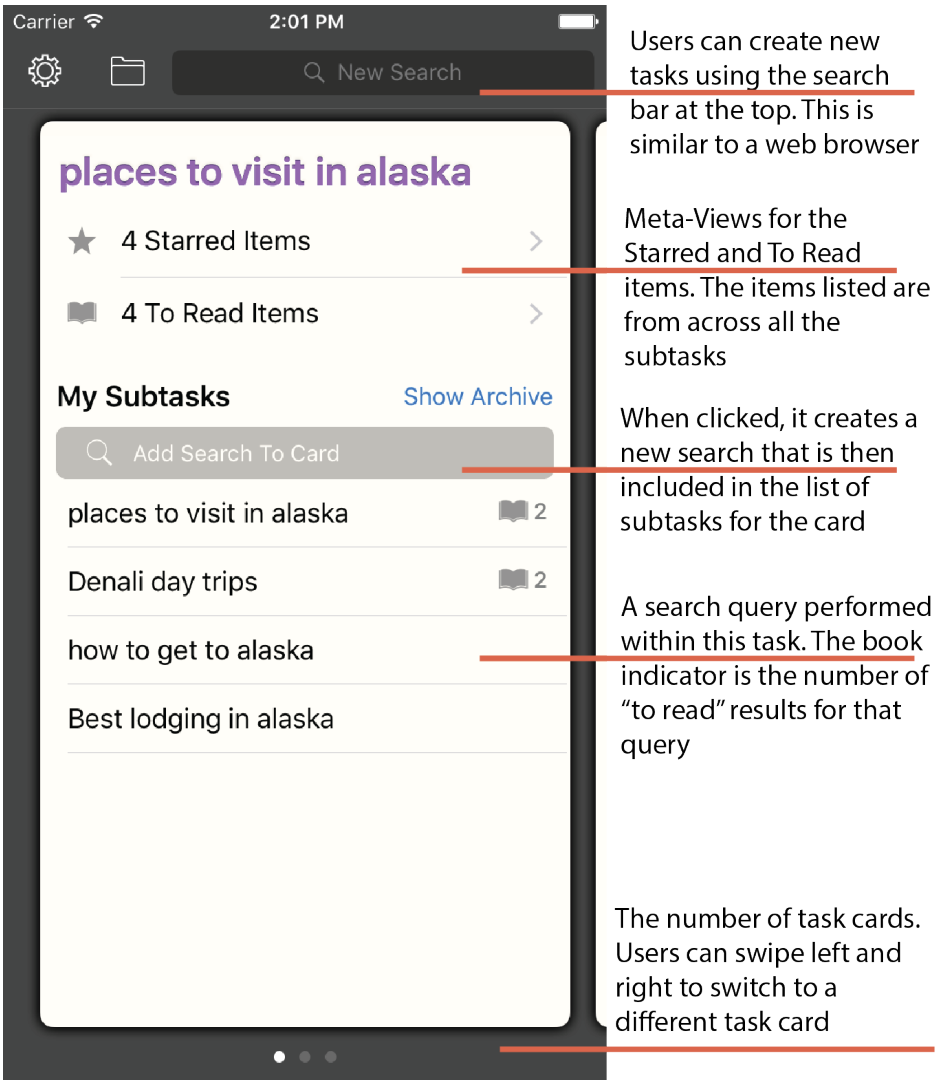


Figure 4.3: The task screen for the task "Places to go to Alaska"

search result are associated with that result. In a normal, tabbed environment, there is no obvious connection from a new tab to a previous page. Even the tacit relationship of being next to each other can be broken if tabs are opened in between. In Bento, there is a fundamental connection between each page and the search result it was opened from. If a user is reading a page that is a deep link from the search result, and they return to the list of results, that page is surfaced under the result with a small clock icon, representing it was a page the user was reading and paused. Similarly, if a user stars, or marks a subsequent page as to read later, it appears in the search results list under its parent search result (see Figure 4.4). This provides an easy way for a user to resume their progress even from a page nested deeply within a search result.

4.3.2 Managing Sensemaking Tasks

Bento not only assists with managing the information from one searching task, but also gives users a way to juggle multiple information seeking tasks at once. Bento features

a separate, second interface for managing the higher level tasks users are working on, and their multiple sub-components. Noting how users utilize tabs, this management interface needed to allow for quick switching between tasks, act as a reminder to resume a task, and create separate workspaces for each task. Inspired by the previous work [60, 73, 147], we decided to utilize an activity-centric management interface based on a task-subtask hierarchy, with the search queries acting as the task unit [111]. However, instead of using this hierarchy as a post-hoc way of revisiting and organizing tasks, we have users actively build their tasks in this structure.

The tasks and subtasks are organized into cards, designed to give the user a quick overview of the current status of their sensemaking activities (See Figure 4.3). In order to make the structure as lightweight as possible, when a user creates a search based on a new topic of research, we create a new task for them, naming it the title of their search. If a user adds a search to an existing task, a subtask appears under the task, with the query as the subtask's name. These task cards allow users to actively switch between tasks using a simple swipe, and their isolated "card" presentation provides the user a clear and present workspace for organizing, and quickly resuming their tasks activities. This is dissimilar from a tab environment, where everything is presented in a flat, undifferentiated structure.

To assist with the tedium of reorganizing tasks, the cards are automatically reordered based on recency – similar to adaptive human memory [8]. When an activity is performed in a task, it is pushed to the top of the list, and older ones never resumed slowly fall to the bottom. The subtasks within a task card follow a similar ordering. More recent queries, shown towards the top, serve as both a reminder for users about subtasks they need to complete, and allows users to orient their work chronologically as their understanding of the information changes. The temporal organization also allows users to scroll down their list of searches so that they can gain a retrospective understanding of how their mental model has evolved over time and restore the context they had when they stopped the search. This structure removes the ambiguity from tab ordering, creating a consistent interaction for later resumption [117].

To create a new task, a user types in a new information need (query) into the top search bar. Initially, we required users to create a task card, and then used it's naming as the information query. However, a normal browser provides a one touch experience in order to create a new search, with a large target for initiating the search. Recognizing that our initial approach broke the user's mental model of searching, we modified the workflow to use a more traditional search bar. The search provides results / suggestions for existing cards, subtasks, as well as normal autocomplete results.

We utilized a similar approach for creating subtasks (or sub queries). Each task card features a separate search button titled 'Add Search To Card' modeled in the appearance of a normal search bar. This reduced the cost for creating a new search to a single tap on a large salient target, in comparison to needing to tap a small button to create a card then subsequently search. On the task card, we provide rich information about the status of the individual subtasks. Beside each subtask, we note the number of starred and to read items from the search results. Besides the ordering of the subtasks, these provide the user with information on the completion of each subtask, as well as the usefulness of that particular information query.

Initially, we required users open up their individual subtasks to view important

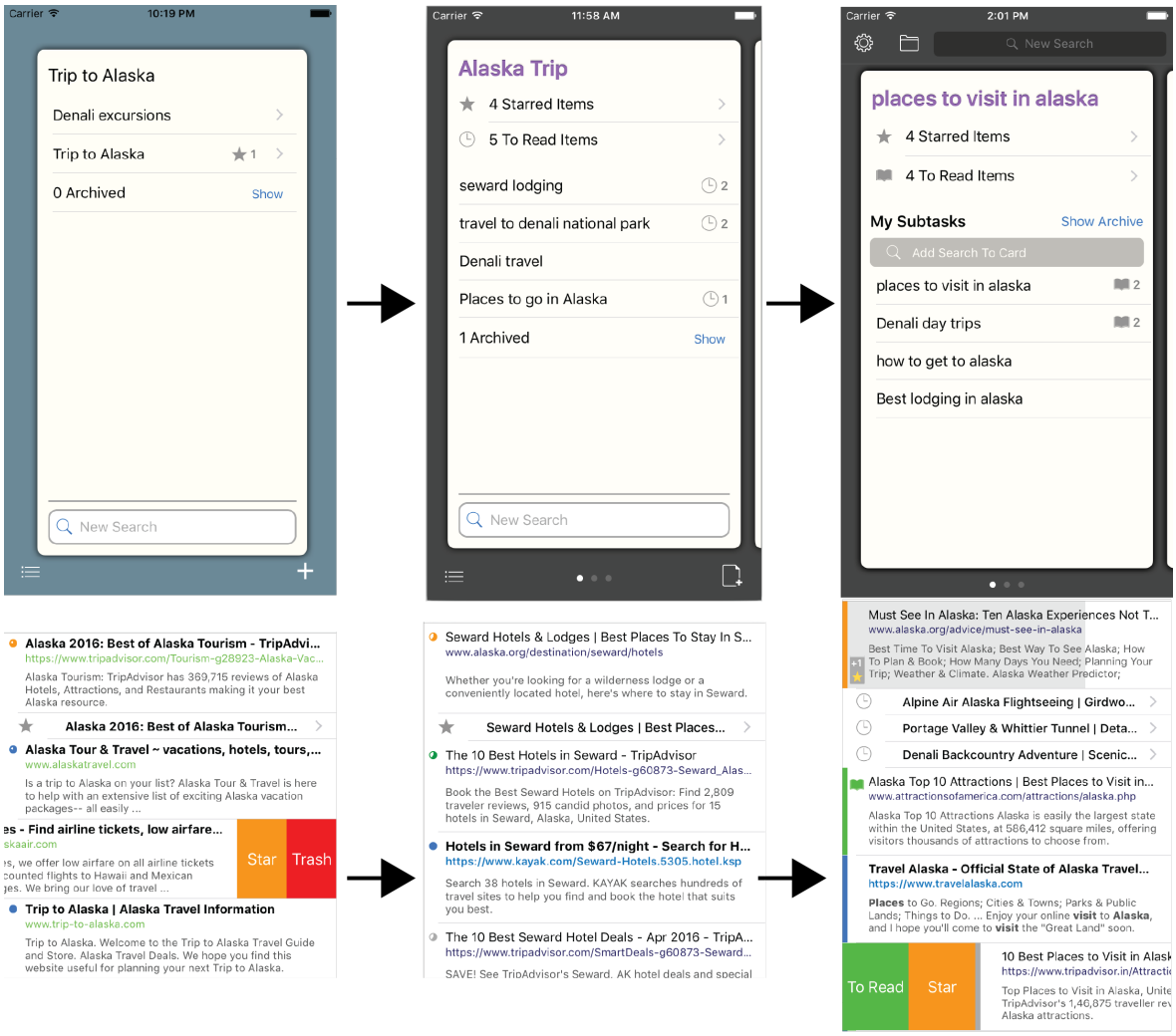


Figure 4.4: The progression of the Bento Brower design. From left to right: Study 1, Study 2, Study 3

saved information, or to make progress. After the first study, one participant noted that "it would be nice ... to see all of my starred items in one place for easy reference." As a result, we added summarization lists on the to-do list card view (see Figure 4.4). These summarization lists allowed a user to immediately look at all of their to read results and starred results across all the searches in a task. The "to read" summarization list became a reading list for the task, while the starred summarization list as a collection of the most important information an individual had collected for the task. These views serve as a way for users to get an understanding of either the important information they have collector for a task quickly, or what information they need to process next in a task.

4.4 Implementation

The Bento Browser application was built for the iOS platform and was available to participants running iOS 8.0 and above. The application utilizes Google's Firebase real-time database and analytics platform to collect telementry data from users. We utilize the Bing API to fetch search results for queries made by users.

4.5 Evaluation

We completed three studies to provide converging evidence on the value of our approach: a lab evaluation, a qualitative real world deployment, and an expanded quantitative deployment. Between studies we used the feedback to iterate and refine the design of the system. We explore whether our dual interfaces of task management and an information triage workspace are able to more effectively accomplish what tabs try to. We specifically look at the pressure points caused by a tabbed interface: organization of tasks, a workspace for information processing and sufficient context for quick resumption of activities. In each of these studies, we optimized the design to promote maximum motivation to actually work on the complex searching tasks by having users work on their own tasks. Rather than trying to evaluate an outcome from a fixed search task, we wanted to capture how individuals found Bento to be useful for a variety of complex searching tasks.

4.5.1 Study 1 - Understanding Triage

The goal of the lab study was to evaluate our approach while controlling for differences in the complexity and nature of the tasks that users engaged in, which would otherwise vary in a field trial. We focused on the triage interface in this study, having users only work on one task while in the lab. It also provided an opportunity to get feedback and iterate on the system's features that might otherwise cause critical issues in a lengthier deployment.

In order to make the lab study task realistic and providing internal motivation, we collected multiple real search tasks from each participant and randomly assigned one to the Bento condition and another to an environmentally valid control, for which we chose the Safari mobile web browser (the default tabbed browser on the iPhone). The study was a within-subjects design in which participants used both browsers (counterbalanced across participant) and provided feedback on them.

Procedure

We recruited 22 participants through a local behavioral research participant pool. Participants ranged in age from 20 to 59, with the majority of participants being local undergraduate and graduate students. 10 participants identified as male and 12 as female. We required that study participants own and use an iPhone to ensure they would be familiar with the existing iPhone operating system and Safari browser. All participants were provided with an iPhone 6s with Bento preloaded onto for use during the study.

Since a single fixed search task might not provide internal motivation to every participant [89], we instead elicited four search tasks of potential interest from the participants themselves during a prescreen. We selected two of these searches based on how participants rated the topics across 4 scales: their knowledge of the topic, the importance of the topic, the expected research time to learn about the topic, and the estimated number of web pages they would have to visit to fulfill their information need. To select one of their proposed searches, we required participants mark it as at least moderately important, have less than a moderate amount of knowledge about the topic, the search would take at least several hours, and the search would require least 8 different web pages. For each participant, we selected two searches that met the criteria (if more than 2 met the criteria, we chose those with the highest values) and randomly assigned them to either the Safari condition or the Bento Browser condition. Some example searches include "How to create an Android application" and "Advice on how to enjoy being a tourist in Japan".

Participants were asked to search for 20 minutes using either Bento or Safari, then to switch to the other search with the other tool (with order of browser counterbalanced across participants). For the Bento condition, before they began they were provided with a brief tutorial that walked them through the interface and features; all participants were already highly familiar with Safari from their own phone use. After completing both searching tasks participants completed a post-survey about their experiences. The survey asks participants to directly compare their experience with the Bento tool to the Safari mobile browser, as well as review some of the features of the Bento tool.

Results

Overall, we found that participants appreciated the Bento interface, finding that compared to Safari, it helped keep them significantly more organized ($M = 4.25, 95\% \, CI[3.91, 4.59]$), and would be more useful for helping them restart where they left off ($M = 4.15, 95\% \, CI[3.66, 4.64]$). Despite participants' high familiarity with Safari, we did not find significant differences in the ease of search creation using our tool, nor did individuals feel less effective using it. Participant did note that Safari was much easier to learn (85% of the participants stated this), however 70% thought that Bento was more helpful in finding pages. This was especially notable considering the prototype status of the system during Study 1 and the additional steps individuals had to go through in order to make and organize their searches.

The comparison was also well supported by feedback on the individual features of the search. On average, participants reported in our post-survey (using 7-point Likert scales) that they enjoyed the software and the features provided by it ($M = 4.95, 95\% \; CI[4.18, 5.71]$). They thought that Bento Browser amplified their search effectiveness on a mobile phone ($M = 5.25, 95\% \; CI[4.49, 6.01]$), they felt confident searching using the tool ($M = 5.05, 95\% \, CI[4.30, 5.80]$), and they reported wanting a tool like Bento Browser for searching on their mobile phones ($M = 5.15, 95\% \; CI[4.37, 5.93]$).

Of all the features, participants found starring pages to be the most useful tool (over 90% reported starring being moderately useful). When asked more about this, they noted that starring pages made it "incredibly easy to save pages" and in general "it was easy to collect a large amount of relevant webpages to read and delete the irrelevant

Question	Study 1 Mean	Study 1 CI	Study 2 Mean	Study 2 CI	Study 3 Mean	Study 3 CI
Which tool did you like better	3.15	[2.45, 3.85]	3.125	[2.18, 4.06]	3.01	[1.94, 3.89]
Which one was easier to create new searches in?	3.4	[2.82, 3.98]	3.126	[1.99, 4.26]	3.38	[2.76, 3.99]
If you wanted to keep searching later, which tool would be better for picking up where you left off?	4.15*	[3.66, 4.64]	4.25*	[3.38, 5.12]	4.44*	[4.05, 4.83]
Which tool makes you feel more at peace?	2.9	[2.16, 3.64]	2.63	[2.01, 3.25]	2.69	[2.05, 3.32]
Which tool makes your information more organized?	4.25*	[3.91, 4.59]	4.13*	[3.43, 4.82]	4.25*	[3.89, 4.61]
I felt more effective using:	3.2	[2.56, 3.84]	3.125	[2.18, 4.06]	3.01	[1.94, 3.89]
It was easier to refind information with:	3.47	[2.96, 3.99]	4.13*	[3.30, 4.95]	3.31	[2.65, 3.98]
I felt more confident that I didn't miss any important sources of information with:	3.0	[2.39, 3.61]	3.38	[1.96, 4.78]	2.53	[1.89, 3.31]
* Significantly different based on 95% Confidence Interval						

Table 4.1: The direct comparison questions were asked on a 5-point likert scale. A higher score indicates preference for Bento Browser, while a lower score indicates preference for the Safari browser. A score of 3 indicated no preference for one over the other. This table covers Studies 1, 2, and 3.

ones." This suggests that the triage interface made it easy to quickly sort through the search results, and persist the important information for later use. A participant directly agreed with this, stating "... I could refind my pages for future viewing. This is very useful for searches that I am more likely to come back to."

4.5.2 Study 2 - Task Management

We iterated on the initial version of Bento based on the feedback from the previous study. Several participants noted that the interface was "clunky compared to the web browser" and it needed to be more attractive. A few others were confused by some of the interactions, such as what happens when they trashed a search result. From the qualitative feedback participants provided in the lab post-study questionnaire, we worked on three areas for improvement: visual attractiveness, better feed-forward and feed-back cues, and the summarization views.

In order to better evaluate the utility of the iterated version of Bento in a more ecologically valid setting we conducted an exploratory field study, in which participants used Bento daily for a period of 4 to 6 days.

Procedure

We recruited 8 participants from the same local participation pool as in Study 1. We required that participants own an iPhone with iOS 8.0 or above installed and had not participated in the previous study. Participants ranged in age from 18 to 24, with four identifying as male and four as female.

Individuals installed the Bento Browser application on their personal mobile device in the lab, completed a short tutorial, then spent 15 minutes working on a search of their choice in the lab so that they could ask questions and get used to the tool. They were then were instructed to use it for at least 10 minutes each day. The application provided a reminder three times a day if the individual had not yet used it for 10 minutes that day. Aside from the time requirement, we did not instruct the individuals to utilize the application in any particular way. We were interested in knowing how individuals used the different features of the application, and which features were the most useful to each individual.

After a 4 to 6 day period, participants returned to the lab for an interview and to complete a post-survey. During the interviews, we asked participants to walk through their usage of the application, showing off any of the concrete tasks that they did, as well as exploring their individual queries. This probe was designed to help users ground their experience of using the app in the specific tasks that they were performing. After the interview, participants completed the same post-study questionnaire as in Study 1.

Results

Post-survey results were very similar to the results from the lab study, with participants significantly preferring Bento over Safari for the questions "If you wanted to keep searching later, which tool would be better for picking up where you left off?"

($M = 4.25, 95\%\ CI[3.38, 5.11]$) and "Which tool makes your information more organized?" ($M = 4.125, 95\%\ CI[3.43, 4.82]$). Additionally, participants also preferred Bento for the question, "It was easier to refind information with (Bento Browser)" ($M = 4.125, 95\%\ CI[3.30, 4.95]$) in favor of Bento. No other questions showed significant differences. Feedback about Bento was similar to the lab study survey, except more individuals cited a desire for a desktop companion ($M = 5.25, 95\%\ CI[3.72, 6.78]$), suggesting that additional usage in different contexts incurred the desire to switch between devices with different characteristics.

The interviews provided further insight into how individuals used the application for their own needs. Participants used Bento for a variety of exploratory tasks, ranging from learning about gardening techniques to product comparison to learning about political candidates. Several of the participants brought up Bento's value in capturing their mental model during the search process, which helped them get an overview of their search and suspend and resume it more easily. P7 specifically noted that you could "see everything that you Googled ... in a straight sequence." and the different triage lists let you "archive what you were thinking about in a single moment ... it was like a screen shot of what you were thinking about." P5 mentioned that he enjoyed "just being able to quickly look at the task list and know what to do next".

Perhaps the most important value perceived by participants was in how Bento structured searches into organized workspaces in which they could make progress. Organizing searches into tasks and recording searches as subtasks were rated highly in the survey (5.5 and 5.9, respectively). This led to some unexpected benefits, such as one participant noting "how easy it is to compare prices this way rather than with a traditional browser". Participants seemed to actively want to keep their subtasks organized, with 6 of 8 mentioning that they enjoyed utilizing the "trash" feature to throw out irrelevant results. We found this interesting because eliciting explicit feedback from users about search results is traditionally challenging, as users could just skip over the search result without having to put in the extra effort to trash it. One interpretation of this is that when users perceive the search results screen as a workspace rather than simply a launching pad they are more willing to invest effort into personalizing it.

Consistent with this, all participants utilized either the starring feature or the to-read functionality. Some of them (P2, P3) indicated that they weren't sure what the point of the to-read functionality was, since they would just immediately read a web page and star it if it was good. In contrast, P1 thought that the "to-read" functionality was one of the most useful features of the application. P1 cited that the feature allowed her to "cue up what she wanted to do next", effectively creating a future list of information to absorb. In a similar vein, P4 created several subtasks at once, and didn't visit them immediately. This allowed her to "just record all of the things she might need to think about for her trip ahead of time, and then just come back to them later". This prospective task encoding was a unique and unexpected benefit of the ability to structure sets of searches together.

Finally, transforming the search results into a workspace made some participants feel a sense of stability and organization; P5 specifically noted that he "like[d] that the results froze from when you went to them" unlike when you traditionally requery a search. These results suggest that a key benefit of the approach was being able to organize and evolve their mental model through a relatively stable workspace.

Participants were also queried about how the mobile form factor of the application either enhanced or detracted from their experience. All of the users (6) who noted something positive about Bento cited the convenience and portability of the application. For example, P3 noted "The ability to search whenever I wanted to. ie waiting in line for something". Two of the users didn't cite anything positive, saying that they preferred larger screens and physical keyboards.

We also noted a number of challenges that users faced with Bento. Some were relatively straightforward, such as confusion around the progress indicator, leading to redesigns for Study 3. However, some were more substantive issues for the general approach. Some participants found Bento useful for complex searches, but overly high overhead for simple informational searches, suggesting that they would like "being able to toggle on/off the organizing part" or "not hav[ing] to create a new task for simple searches which would not require detail planning and organization". Another common complaint about mobile phone searching more generally was the difficulty of typing, e.g., "typing on a phone screen can be arduous." Exploring the tension between low overhead for simple searches and supporting complex searches – especially when the former can transform into the latter – may be a fruitful area for further research.

4.5.3 Study 3 - Behavioral Traces

Studies 1 and 2 provide converging evidence about the value of Bento's two interfaces of task management and a triage workspace. However, although the field trial in Study 2 provides suggestive evidence and scenarios of how participants used Bento, one weakness is that it relies on self-report data which could be biased or incomplete. In order to collect richer quantitative data about Bento's usage in the field we conducted a third study in which we instrumented the browser with data collection capabilities and analyzed participants' actual usage data. This also gave us an opportunity to iterate on the design to address the issues discovered in Study 2, e.g., confusion around the progress indicator and lack of support for quick, simple searches.

Again, with the previous study, we performed some modifications to Bento's appearance based on feedback. We focused on improving the readability of the search results and improving the learnability and "first use" experience. We introduced a more coherent first use scenario, adjusted the progress indicators on the search results screen to their final form, and modified task and subtask creation.

Procedure

Study 3 followed the same procedure as Study 2 but with the updated Bento application, with more participants, and for a longer period of time (10-13 days). Utilizing the local participation pool, we recruited 16 participants with ages ranging from 18 to 25. Participants who participated in the previous field study or lab study were not eligible to participate. Five participants identified as male, and 11 as female. Twelve of the participants were undergraduate students, while 4 of them were graduate students. Afterwards participants completed the same interview as the first field study, and completed a slightly extended version of the questionnaire.

Results

Survey results and feedback were overall similar to the first field and the lab study, with significant preference towards Bento for the two questions: "If you wanted to keep searching later, which tool would be better for picking up where you left off?" ($M = 4.44, 95\%\ CI[4.05, 4.83]$) and "Which tool makes your information more organized?" ($M = 4.25, 95\%\ CI[3.89, 4.61]$).

The key research question for Study 3 was quantitatively investigating whether participants were in fact managing complex searches and utilizing the different features of Bento. Each individual created on average 13 tasks (M= 13.06, SD= 8.433) with on average 3 subtasks (M= 3.13, SD= 1.48), suggesting that participants were indeed engaged in complex searches with multiple subtasks. There was high variability between users, with some users having as many as 14 subtasks within one task. Drilling down further, for each subtask participants opened an average of 7.7 pages (M= 7.7, SD=5.41), suggesting that they were engaging in tasks that involved significant exploration. To check this against participants' own perceptions we asked them to classify their searches as either complex or simple when they came back into the lab at the end of the study. Participants classified 35% of their tasks as complex searches, suggesting that they were engaged in complex searches but also using the system for simple searches, addressing an issue raised in Study 2. Participants found value in Bento's organization and resumption capabilities for complex searches including researching "fandom", bus routes, and radiation oncology internships. For example, one participant explicitly mentioned "I learned the sort of tasks that bento is good for – [it] requires several (subtask) searches. for e.g. transferring money internationally there's wire transfers, exchange rates, foreign check processing fees, different charges for diff banks."

The mobile form factor in this longer study offered some surprising and unexpected benefits for the sensemaking process. One user mentioned that the mobile versions of web pages were actually easier to parse: "Many result pages are mobile optimised, such that the content delivered may be more condensed and the design of the webpage more minimalistic." Another user cited a scenario where it is actually impossible to have a laptop – cooking in the kitchen. In this case, her mobile phone is the only tool she can use to perform sensemaking: "When I'm cooking and I have a recipe loaded, I will prop up my phone on the counter. My laptop would take up too much space."

Participants consistently used many of the features of Bento. Individuals reopened subtasks on average about 2.2 times, starred 7.05% of pages visited, marked 5.84% as to read, and trashed 4.24% of results (note only 20 results were loaded at the time of the search). Each individual had approximately 23 sessions over the study period, so about 2 application sessions per day. When asked what feature they liked most, participants mentioned the organization of tasks and subtasks (9); being able to come back to searches (3); marking pages to come back to later (2); the gray background progress bar (2); starring pages (1); and the overall design (1). When asked what they would like to change most there were a large variety of suggestions, most having to do with not having the features of a full search engine like Google they were familiar with (e.g., access to google scholar or images, answering questions directly after a search, having better search results). Two participants mentioned "quick search" as desired, suggesting there may still be a need to support simple searches more easily than in the

current approach.

There was high variability around the use of features and types of searches participants engaged in, with some focusing on simpler searches and some more complex ones. To examine whether the type of search affected perceptions of the tool, we correlated the ratio of complex:simple searches with perceptions of the tool from survey responses. Those with a higher number of complex searches:

- Liked Bento better than a mobile web browser ($r(14) = 0.638, p < 0.01$)
- Felt more at peace using Bento ($r(14) = 0.71, p < 0.01$)
- Felt more organized using Bento ($r(14) = 0.55, p < 0.05$)
- Felt more effective with Bento ($r(14) = 0.834, p < 0.01$)
- Wanted to keep Bento on their phones ($r(14) = 0.644, p < 0.01$)
- Felt Bento improved their effectiveness on mobile phones ($r(14) = 0.65, p < 0.01$)

4.5.4 Summary

Across the above three studies, we found evidence that users appreciated both the task based organization interface, as well as the search results workspace interface. Together, these consistently made users feel more organized and feel like they could resume their activities more easily. Based on these findings, we also have a couple of additional takeaways.

Users appeared to use a few strategies, aligned with many goal activation theory approaches [4]. For example, in study 2 P4 noted that she queued up searches for later exploration, largely a prospective planning task. Conversely, we had individuals like P2 and P3 who didn't really understand the to-read feature, another planning tool we had incorporated into our design. This suggests that different user populations might practice different planning techniques for their exploratory searches, and while the structure appeared to be amenable to most of them, having tools for both planning and retrospective recounting could be essential to the design of these systems. This information was similar to what was found in Study 3 with user preference for different features of the system. Most users liked the overall organization for easy re-finding, however some users liked the specific planning features, such as the to-read feature and the grey progress bars. Supporting both of these resumption use cases will be key proceeding forward.

We had several complaints from users about the overhead of Bento for simpler searches. In both studies 2 and 3, individuals noted that they wish they didn't have to make an entire task card for just simple searches. However, in some of our interviews, individuals noted that their simple searches, such as looking up an actor in a movie, often blossomed into more complex searches, such as looking at what else that actor was in, what roles they typically play, etc. Having a low overhead, while also supporting this transition of simple search into complex search was an issue in Bento that was not entirely resolved.

4.6 Discussion

With Bento, we introduced a novel way to manage complex searching tasks on mobile devices. Bento creates a scaffolded process that separates the task management and workspace functions of tabs into two distinct interfaces. Its two focused interfaces provide users with the necessary affordances to make progress on their sensemaking tasks even within the constraints of mobile devices. This structure is able to meet the complex demands of sensemaking and mobile work, and can be used for later transfer, hand-off, and resumption.

From Bento, we were able to discover a several important benefits regarding a scaffolded task-based structure. First, functional units that can be leveraged in future work. For example, consider the goal of making sensemaking independent of person. The tasks present in Bento could serve as the key unit of collaboration. Since they represent a specific, independent information goal, a task could be shared and worked on by a team of individuals. Subtasks could be delegated out to certain individuals to explore, and because all of the information is tracked, mechanisms such as the star feature could be expended into a voting feature. Similarly, the task could be handed-off to another person. The details about which pages were found important and which queries were used could provide a valuable starting point to another individual researching the same topic [52].

Along another line, the structure could be used to allow for easy transfer and resumption from other computing systems. Indeed, one participant (P6) noted that he wished there was a web based version. He utilized a number of different devices, many of them not his own (such as those provided by his university). Having this tool available on any platform would let him pickup his searching or find some information that he needed. For example, a similar approach could be instantiated as a virtually identical desktop browser that syncs with the mobile version. However, moving to the desktop may provide other design opportunities given the additional screen real estate; for example, the three level hierarchy (task > subtask > page) of Bento on a smartphone might be flattened to two levels (e.g., task cards and a subtask pane of search results with a reading list, similar to an email application) or even a single level (by incorporating the task card into the view). These changes would keep the integrated tab management and exploratory search support of Bento while being a more efficient way of reading and exploring. Better support for text entry and annotation on desktops could also benefit a future version of Bento on the desktop.

Finally, it is possible that using an approach such as Bento may change the strategies that people use in search [110]. For example, although we expected users to add subtasks as they encountered new information, one participant found it useful to do the opposite: "my searches were more focused because i tended to brainstorm at the start of a task and added subtasks at that time". Essentially, the list of searches in Bento were serving not only as a history list, but also as a list of ToDos that needed to be accomplished. While we didn't initially design for this behavior, it ends up being highly congruent with the dataflow-centric sensemaking model (See 1.1).

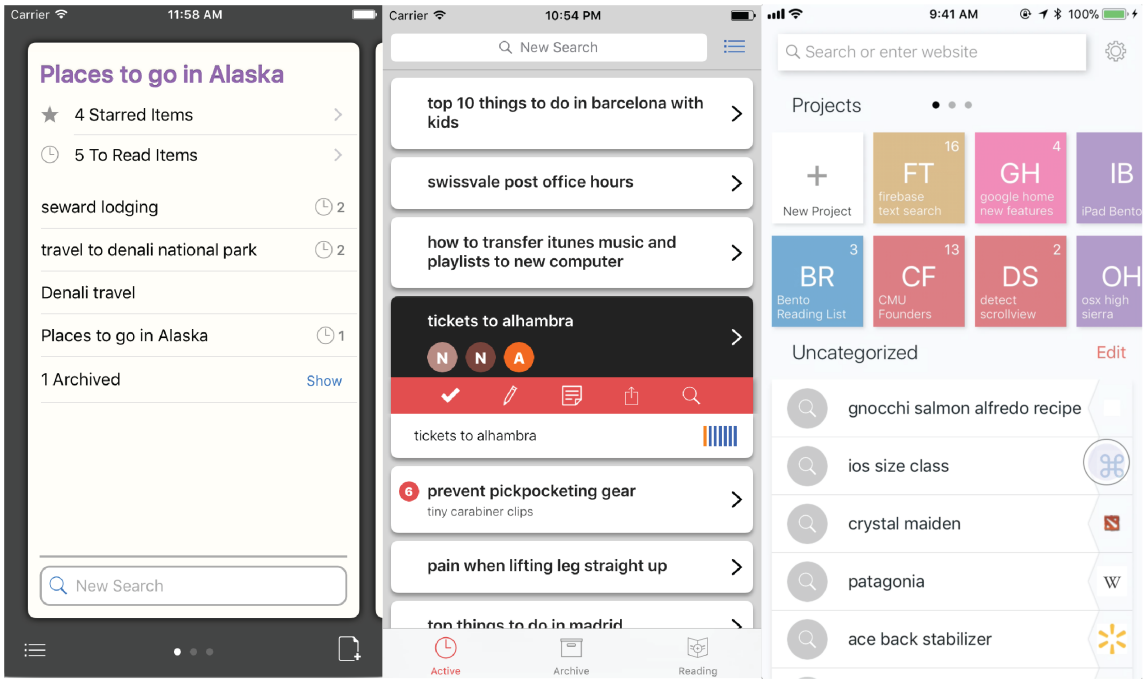


Figure 4.5: Three different iterations of Bento. The left-most was the final iteration tested in the above studies. The middle version is an intermediate prototype that allows users to stack searches together to form a project. The right-most version is the final, deployed version of Bento where users see all of their recent, uncategorized searches in the bottom half, which they can drag into either existing or a new project in the top half

4.7 Bento Iterations

While we only tested the initial version of Bento to gauge the effectiveness of a search-based task scaffold for source foraging, we realized that the interface had limitations. These centered around supporting simpler information seeking tasks, and their transition to sensemaking tasks in a more fluid manner. We went through several additional iterations of the interface (Figure 4.5) in order to refine some additional interactions in support of this: post-hoc task creation, a separate queue of recent simple searches, and quick access to the last search result page viewed for a search.

First, rather than requiring users to create task cards up front in order to perform and track searches, users could create a new search instantly from the homepage of Bento. These searches would then be tracked in a "recent searches" list, which would be shown below the list of projects the user had. Users could then take one of these searches and drag them into a task in order to persist them there, while also giving them some more task-oriented features present in the original version of Bento. This post-hoc task generation was designed to further reduce the barrier required to start a simple search task — one of the most common browser interactions — but still allow users to transition those tasks from simple searches into a more complicated project

with progress tracking. Recognizing that simple searches are often used as either purely navigational or to answer basic questions, we instead introduced a button on the right hand side of these search queries that would take you back to the most recent page you've visited for that search. This would allow for the resumption or quick refinding of simpler activities, giving users the ability to decide between

Chapter 5

Siphon: Flexible Collection

While Bento gave users the ability to triage information at the source level, users are unable to effectively break apart and triage sections of an information source. Throughout the sensemaking process, users have varying levels of uncertainty [28], resulting in them often selecting large sections of information early on in the process, and smaller, key sections later. To tackle this issue, I developed the Siphon toolkit, which allows users to use a variety of selection interactions to annotate and extract information ranging from an entire page, to a specific word. Uniquely, Siphon also maps any selection to the underlying HTML of the webpage, which allows the toolkit to pass along the underlying content, highlight and maintain a connection to the selected content, and rerender the selection in other contexts.

5.1 Introduction

Saving online information for further use, later reference or to share is an increasingly common activity [97, 105, 135]. Estimates put the current amount of time people spend online at almost 24 hours a week – up from 9.4 back in 2000 [96]. As users spend more an more time online across a myriad of devices, researchers have created a number of different tools for helping users leverage this content. Efforts have focused on improving content selection [29, 57, 129, 155], simplifying or enhancing information extraction [47, 64, 132, 133] and supporting content organization [24, 66, 160].

While at first glance these tools may appear to have little in common, by approaching them as active reading and sensemaking support tools [124, 130], they all share a common core annotation workflow that must be present for them to be functional: a way to select content, a means to extract that content, and then some way to reuse or consume that content. Different tools enhance different parts of the workflow, however they then must rely on very simple inbuilt implementations (e.g. Thresher [64], an extraction focused tool, needs to rely on the built in browser selection tool) for the other portions of this workflow. Supporting this workflow is further hindered by the ever increasing complexity and diversity of web content. Difficulties include accessing the underlying content when performing selection, identifying and extracting the appropriate content in a way that is portable and displaying information to users in a way that is recognizable, interpretable, and traceable. These issues are largely caused by the way web content is

rendered – there is a one-way process of converting the underlying HTML content into the final visual representation seen by end users. It is then further compounded by the number of libraries that augment this rendering process, such as React, where the final output is largely driven by Javascript, rather than just pure HTML and CSS.

In order to enable developers and researchers to quickly build the next generation of tools for accessing and saving web information, we designed Siphon, a toolkit for web annotation. Siphon provides support for the entire core annotation workflow through a modular, extensible interface while also providing a set of feature rich defaults for when developers choose not to augment a specific workflow stage. Inspired by WYSIWYG (What you see is what you get) design it provides developers with a simplified, abstracted mapping between the rendered output of a web document, and the underlying DOM model driving that output. This is instantiated in a few key ways:

- Siphon provides a unified interface for managing multiple types of selection on a single page, and simplifies the definition of a selection technique into a condition that must be true to be in that selection mode, and then three lifecycle callbacks that manage the setup, update and teardown of any elements that support that selection style.
- Content can be specified for extraction through a browser range object, a set of DOM nodes, or, uniquely, a set of window coordinates. In the coordinate case, Siphon will convert those coordinates back to the original DOM nodes generating the underlying content.
- The output of the annotations are graphical, interactive, and retain their connection to their original content. When an annotation is made, Siphon creates an self-contained, styled, interactive HTML snippet that can be shown in a standalone context. Additionally, it records the XPath of the original content, allowing developers to trace and surface already saved content.

Through these features Siphon allows developers to both create web-read annotation tools and techniques faster, while also maintaining a high level of support for the other components of the core annotation workflow. To further contextualize how Siphon was designed to support the research and development efforts for improving selection and extraction of web content, we next discuss the previous work describing those research areas. Then, we will describe more concretely the Siphon toolkit and the underlying toolkit features. Finally, we describe several possible implementation scenarios for Siphon, both from a more end-user focused developer, as well as a researcher who might use the toolkit for a new selection interaction technique.

5.2 Related Work

Siphon draws its definition for the core components of annotation: selection, extraction, and reuse, from active reading [112, 135] and sensemaking literature [124, 130]. Alder et al in their taxonomy of reading [2] discuss how reading is often followed by the key activities of extracting and integrating information from different sources. Tashman and Edwards [135], evaluating digital active reading, note that tools for active reading

need to support flexible selection and annotation of content, the ability to record and maintain context for annotated content, and the ability to easily visualize annotations and the relationships between them. Similarly, Pirolli et al.'s sensemaking model's [124] foraging loop consists of finding information, reading it, and then extracting it for later organization and use. These models suggest that an effective digital annotation tool should support flexible selection of content, the ability to extract content while maintaining a rich context, the support for further use of the information in a variety of contexts.

In order to build a toolkit that is able to support these three areas, we look at the different requirements and needs of existing active reading support tools.

5.2.1 Selection Techniques

First, we looked at different selection techniques, and considered how they might then might pass their output to the extraction stage of the process. Researchers have generated a number of new selection techniques in response to new device interaction modalities (e.g. touchscreen phones) or the ability to more reliably select and manipulate a particular type of content (e.g. entities). These include improving copy and paste [33], supporting margin-based annotations [155], allowing for navigation and selection on touch devices [57, 129], providing enhanced selection for small targets [51], and allowing for fuzzy selection [29]. Across these different selection techniques, there were some common patterns. First, techniques first had to be triggered by entering a selection mode, usually done by a specific input event on in a particular location (e.g. a bezel [33]) or some type of content (e.g. a click on a selectable entity [51]). Once in selection mode, interactions users performed updated what was being selected until they completed the selection, or they canceled it. Once the selection was complete, the user exits selection mode until another input event triggering selection mode occurs. Therefore, a toolkit could assist developers by providing a simplified lifecycle for defining selection techniques.

5.2.2 Leveraging and Extracting Web Content

Next, we looked at systems that focus on improving the process of extracting content from web, either by assisting users with the process, transforming it into a more usable form. Thresher [64] and the work by Dontcheva et al. [46, 47] assist users by automating extraction of specific fields and content from web pages, and providing a more contextually relevant view for that content. Zoetrope [1] allows users to see the web history of a specific section of a web page using XPaths to align different versions of a web page. Finally, Citrine [133] augments traditional copy and paste by transforming the output into a structured form usable by other applications, such as Microsoft Outlook or Excel. LemonAid [35] supports inline help on web pages by anchoring help to specific web page DOM elements, while Hunter Gatherer [160] and Internet Scrapbook [134] allow users to specify entire web "components" they wish to save, which are then later surfaced as live content sections.

Across these tools, there are a few different common types of content that are being saved. LemonAid, Thresher, Zeotrope and Dontcheva et. al,'s system all have users

point out specific DOM elements they wish to save. While some of these systems might benefit from broader, more flexible selection tools (such as a bounding box for Thresher), others such as LemonAid are about user interactions with specific elements on a page. Citrine relies on traditional browser selection to support enhanced extraction of information. Finally, Hunter Gatherer and Internet Scrapbook, although it uses DOM based selection, is largely a visual snipping tool, and thus would benefit from a graphics-based selection techniques, such as BezelCopy on a mobile device. A toolkit, in order to support content extraction across a wide range of selection styles, should at a minimum support these three different content specification techniques.

5.2.3 Reusing Annotations

Finally, we consider how systems that support unique interactions with extracted content can be better supported by an annotation toolkit. Tools for spatial organization, such as IdeaMache [98], the Sandbox System [154], and WebBook / Web Forager [24] have very visual representation for the content, allow users to see content from multiple sources in one location, and flexibly organize content while maintaining a connection to the original source. Hunter Gatherer [160] and Intenet Scrapbook [134], although more extraction-centric tools, provide a simple overview page for collected content, where the web components collected are shown together and rendered live. WebView [37] and PadPrints [61], more history focused tools, and also utilize thumbnails of content to aid in revisitation of web pages. Piggy Bank [66], on the other hand, utilizes metadata from websites to provide a rich bookmark entry the user can then search for and tag with further details. Across these systems, there is a need for a rich representation of an annotation, be it a picture, a metadata enhanced card, or live snippet from a webpage. Additionally, users need to be able to display this content in a variety of settings, manipulate it easily, and have a link back to the original source of the content.

5.3 Siphon

Using the aforementioned previous work as a guide for developing a unified annotation toolkit, we created Siphon. Siphon provides support for the core annotation workflow of selection, extraction and consumption through three major components: selection definitions, annotation objects, and a store for persisting and restoring annotations. This section breaks down each of these components, and provides a simple example of how a developer can use these to provide enhanced web annotation.

5.3.1 Selection Definitions

A selection definition is simplified event lifecycle for triggering, updating and completing selections (See Figure 5.1). A definition is composed of four main parts: a selection condition that must be satisfied and three lifecycle callbacks a developer can implement. The selection interaction begins when the selection condition is met, and ends when the condition is no longer met (such as the left mouse button must be pressed during selection). At the start of the interaction, the onSelectionStart callback is called, this

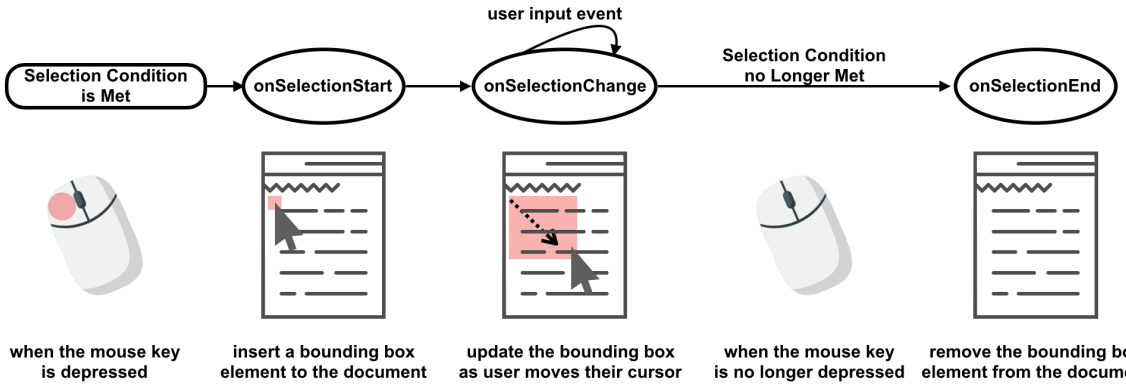


Figure 5.1: Siphon's simplified selector state diagram (Top) and an example Selection Definition of drawing a bounding box for selection (Bottom).

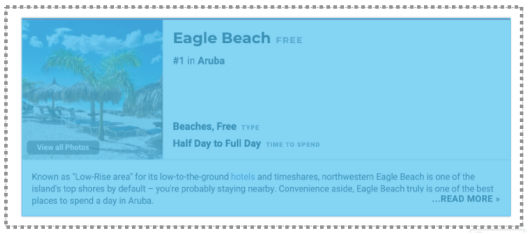
allow developers to setup the selection interaction and any feedback mechanisms. For example, in the case of drawing a snippet box, inserting a bounding box element to the DOM tree at the cursor position. Whenever a Javascript user input event[1] (such a keyboard events, mouse events, etc.) occurs, the onSelectionChange callback is called. This allow developers to update the visual feedback based on users input. For example updating the boundary of the previously inserted selection box. And finally, when the selection condition is no longer met, the onSelectionEnd callback is called, allowing the developer to cleanup the selection interface, and based on if the user completed or canceled the selection action, create the appropriate Siphon annotation objects in response. In the case of the previous example, the developer would make a Siphon annotation object from the final bounds of the selection box and then remove the box.

Selection definitions allow for multiple types of selection interactions on a single page, managed through Siphon's unified event management interface. Traditionally, a developer would have to listen to the individual events associated with their selection, so in the above case the mouse down, mouse move, and mouse up, and then manage the state of their selection based on the ordering of those events. In Siphon, a developer registers the selection definitions they want to use in their application, for example a snippet selection tool, force touch selection, and one click image saving. When a mouse, keyboard, pointer or touch interaction occurs, Siphon takes the event object and merges it with previous events to create a unified event object. This is then checked against all of the selection definition conditions registered with Siphon. If one is found to match, that selection definition is considered active until its selection condition becomes false – Siphon only allows for one mode of selection to be active at a time.

5.3.2 Annotation Objects

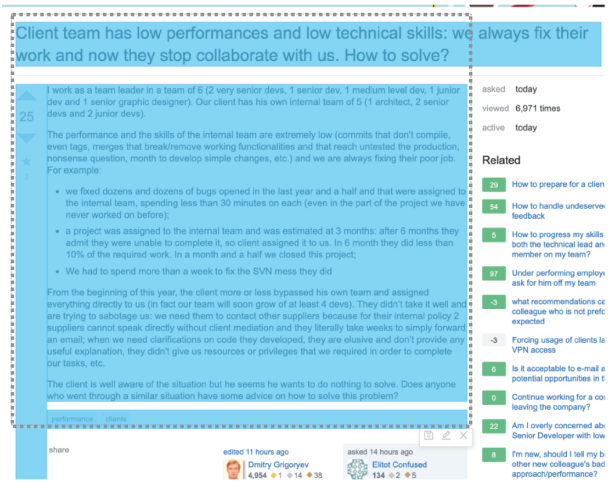
After a selection is completed, a developer can generate an annotation object based on the final output from their selection interaction. Annotation objects on creation identify the DOM nodes selected, extract the appropriate surrounding HTML including style

[1] https://developer.mozilla.org/en-US/docs/Web/API/UIEvent

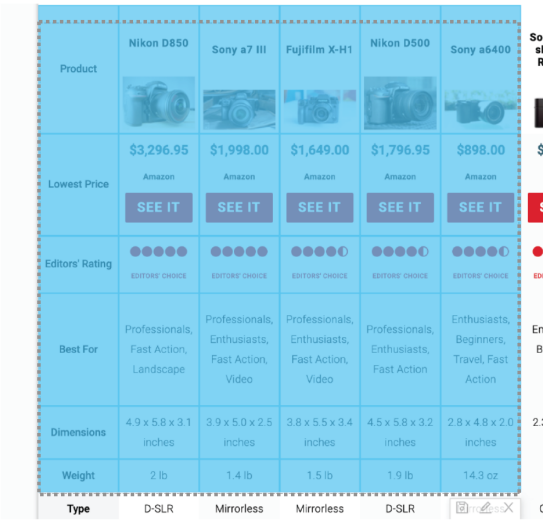


Selecting part an entire webpage component

Selecting part of a webpage component



Selecting multiple pieces of content
(note how Siphon expands selction to capture the full header)



Selecting a portion of an HTML table

Figure 5.2: Examples of Siphon's graphical selection tool. User specified bounding boxes in gray dotted lines are resolved to a set of DOM elements highlighted in blue

properties, and generate a persistent reference to the original content. These objects can then be used in a variety of ways – they can act as a persistent references to the DOM elements on a page, they can be rendered as-is in external systems or interfaces, or the underlying HTML can be mined for metadata or specific content to surface. These can drive tools for consumption – such as visual organization tools, tools for performing in-page markup, and tools for automated extraction of information from pages.

Annotation objects are be generated from three core types of references: a single or a collection of DOM elements, a browser range object (generated through the default browser selection interface), or a set of graphical coordinates (i.e., a bounding box). These were selected based on the requirements of previous selection methods and tools for extracting web page data [64, 133, 160]. The graphical coordinates-based selection differs from the other two selection input in that the bounding box coordinates need to be resolved to its underlying DOM elements. Siphon accomplishes this through a bottom-up traversal of the DOM tree. First, Siphon calculates the intersection of all the block-level leaf nodes in the document with the current selection area. Then, after removing any fixed position elements, it iterates through those nodes to find the nodes with the greatest percentage of area overlap. Until at least one element is found, the algorithm adjusts the overlap threshold to a minimum of 50%. If no elements are found at this point, Siphon assumes the user was selecting a portion of a block level

element (typically an inline DOM element), and will return that element instead. After performing the leaf filtering, Siphon then attempts to reconstruct the DOM tree from the bottom up, searching for parents whose visible leaves are all present in the output of the leaf filtering. This reconstructed set of DOM elements are considered to be the selected set of DOM elements.

Additionally, Siphon allows for specifying a single point, instead of a bounding box, in the case a developer want to anchor an annotation to the page e.g. leaving a manual note attached to a point in a page. In pen and paper interfaces, and digital document annotation interfaces [136], users might accomplish this by leaving a note in the margin near the content they are referencing. However, if a user were to click on this whitespace in a browser, this would most likely reference a large container element on the page, which not at all correspond to where they would actually want to leave a reference. In this case, Siphon assumes that the users is probably referring to some content in the center of the viewport, and uses a 1 pixel tall bounding box across the entire width of the page to try and find that center element. This allows for the reference to be attached to an element in the element in the DOM that is more closely aligned with their reference and follow it appropriately i.e on resizing.

Once an annotation object has a set of DOM nodes to work from, Siphon then works to generate a static HTML snippet that can be rendered in alternative contexts. This is similar to the output of Hunter Gatherer [160], however instead of just creating a reference to to the content, Siphon creates a fully styled, portable HTML snippet. This is uniquely challenging, due to the cascading nature of CSS, as pages are designed to be rendered as a complete entity, not piecemeal.

To accomplish this, first Siphon then creates a copy of the DOM nodes from selection and embeds all of the currently applied CSS styles into the nodes and ensures any external URL references are absolute in nature, rather than relative. Siphon performs several optimizations to ensure all possible CSS styling and HTML content are captured, as well as ensuring the snippet is properly rendered in different sized containers. First, Siphon removes any metadata / scripting tags, and also embeds the content from accessible iFrames into the snippet. Second, if any CSS pseudo elements (i.e. before and after) are present, Siphon extracts those as a separate CSS style description, assigns a unique additional class to their corresponding element, and includes this extra CSS style description in the final HTML output. Finally, in order to properly support layout reflow in different contexts (i.e. different sized containers), Siphon ensures that any CSS styles included utilize their computed, rather than their actual values (e.g. an element with a percentage based height keeps this as its embedded value instead of the currently rendered pixel based height the element would have). In order to reduce the size of the final HTML snippet, Siphon detects any default CSS values, and removes those from the embedded CSS style definition. While this works for many cases, there are usually some subtle layout differences (See Figure 5.3), and in the case of highly interactive content, such as a D3 visualization, the snippet generation can fail altogether.

Finally, Siphon supports referencing the original source of an annotation using a set of XPaths and the document URL. As found in Zoetrope [1], maintaining the position and provenance of the original annotated content is challenging due to the always changing nature of web pages, as DOM elements can shift position in future revisions of a page. Siphon currently uses the open source npm package xpath-dom [63]

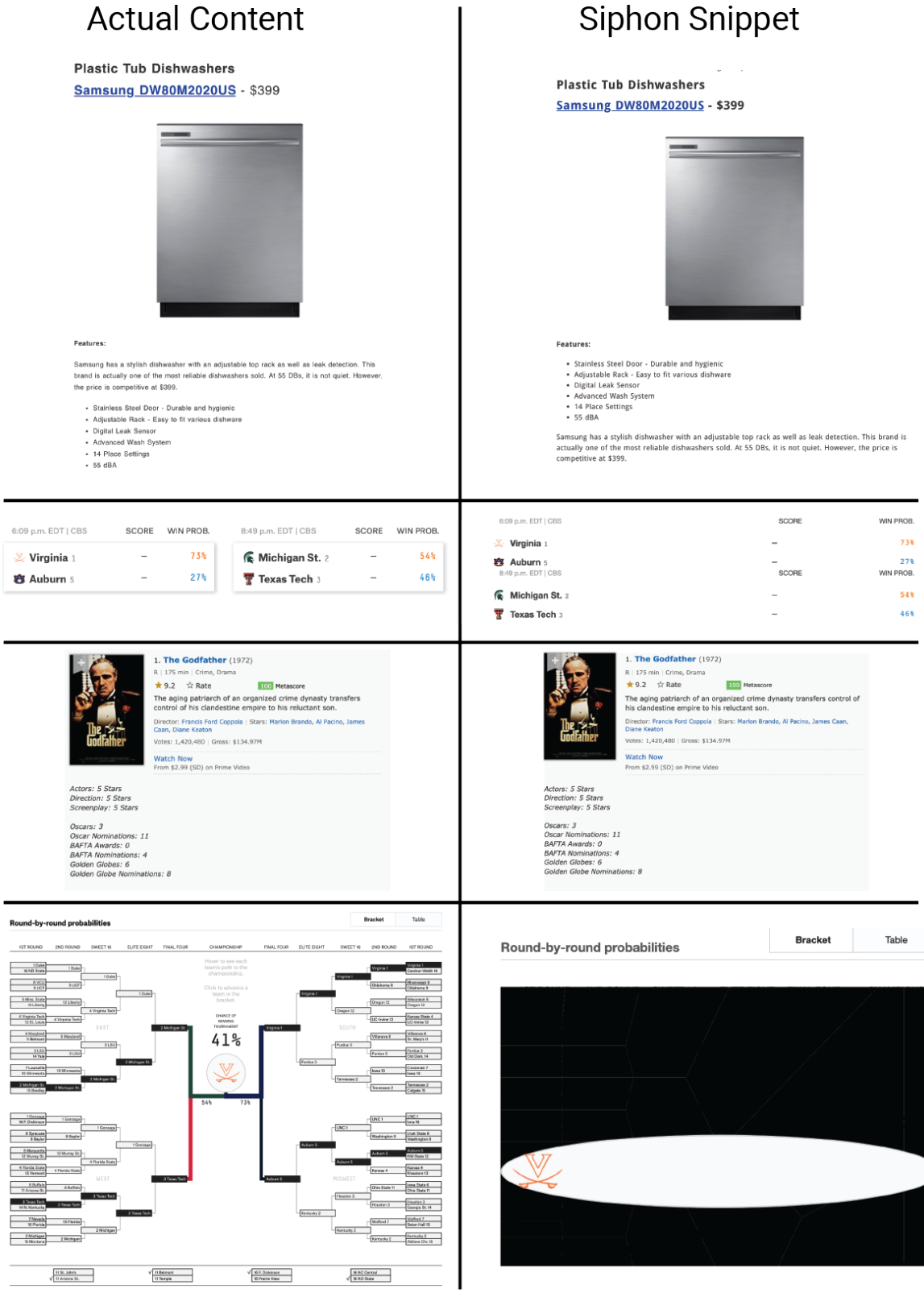


Figure 5.3: A comparison between the actual rendered content on a webpage and an extracted Siphon snippet that can be rendered outside of its original context

to accomplish this, however more sophisticated XPath generation and resolution could be performed, such as that in Doncheva et al.'s work [46, 47].

5.3.3 Annotation Store

The final component of Siphon is a store for all the annotation objects generated by users. The default store will only save Siphon annotations to memory, as it is up to the implementer / developer to decide how and where these annotations will be saved. The store defines 4 CRUD-like core methods for saving, removing, updating and importing annotations that should be implemented by the developer using Siphon. These methods serve as the interface for serializing and deserializing the Siphon annotation objects to/from JSON, as well as, on deserialization, finding the original location an annotation was made on a page and marking that with a custom class. Because modern webpages are increasingly dynamic and utilize a large amount of client side scripting for generating the final DOM content, the store also monitors changes to the current page's DOM and searches for any annotations that have not been properly restored and attempts to restore them.

5.3.4 Implementation

Siphon is implemented as a vanilla Javascript library that can be utilized in a number of different contexts: as a browser extension, as part of an electron application, or embedded into a specific webpage / application. The only requirement is that Siphon has a modern DOM Document object to work with – thus it can also run in a headless browser. The library can be found on github: https://github.com/BentoBrowser/SiphonTools

5.3.5 Built-In Tools

As mentioned, the Siphon toolkit provides a number of built in selector tools that let developers utilize the toolkit without having to manually define a number of basic selection types. These built-in tools consist of:

- **HighlightSelector** Supports the integration of the built-in browser selection tool with Siphon
- **SnippetSelector** Allows a user to draw a bounding box around some content on the page
- **ElementSelector** Users can select a set of DOM elements directly. As they hover over an element, a blue overlay is displayed on the element. They can click on this element to add it to their set of selection elements. If they click on an already selected element, it removes that from the selection set
- **ClickSelector** An implementer passes in a CSS selector to this selection definition, and any click on an element that matches that selector will trigger this definition.
- **HoverSelector** An implementer passes in a CSS selector to this selection definition, and when a user hovers over an element matching this CSS selector, this definition will become active.

- **ListSelector** A more experimental selector that attempts to automatically identify "lists" of elements on a page (these can be DIVs, table elements, etc), and then provide those as a set of possible selection targets. Users, when this selector is active, can then select list elements in a manner similar to the ElementSelector.

Aside from these built-in selectors, developers can also create their own by using the selection definition framework of Siphon.

5.4 Scenarios

In this next section, we then look at how a researcher or developer might leverage and integrate Siphon into their tool in order to streamline the development experience, and enhance the tool's capabilities.

5.4.1 Supporting Selection

First, we look at how a developer wanting to use force touch highlighting [29] might create a selector definition for that selector style. As seen in Code Example 5.1, the amount of code required to enable such a technique is fairly short and self-explanatory, as the technique follows Siphon's selection flow fairly closely.

In this code, the developer defines a condition based on the current touch selection state and the amount of force applied to the screen. As the user varies their cursor position and the amount of force on the screen, the onSelectionChange callback is called, and the developer updates the selection bounding box and, based on the user's horizontal finger movement, decides if they are marking the content to be saved. Finally, once the user is no longer touching the screen – making touchStart and the selection condition false – the onSelectionEnd callback is fired, and, based on the difference in the user's horizontal finger movement, the annotation is either saved or the selection is canceled.

```
1  var saving = false
2  export default function ForceTouchHighlighting()
3    return {
4      conditions: function(e) {
5        return e.touchStart &&
6               e.touchStart.force > MIN_START_THRESHOLD
7      },
8      onSelectionStart: function(e) {
9        createSelectionBox()
10     }
11     onSelectionChange: function(e) {
12       updateSelection(
13         e.touchMove.clientX,
14         e.touchMove.force
15       )
16       if (e.touchMove.clientX - e.touchStart.clientX >
             MIN_SAVE_THRESHOLD) {
17         saving = true
18       } else {
19         saving = false
```

```
20        }
21        updateSelectionBoxAppearance(saving)
22      },
23      onSelectionEnd: function(e) {
24        if (saving) {
25          saveAnnotation(selectionBox)
26        } else {
27          cancelAnnotation()
28        }
29      }
30    }
31  }
```

Code Example 5.1: The pseudo code for implimenting the force touch selector in Siphon

A couple of more complex use cases would include Entity Quick Click [19] and the Click and Cross cursor [51]. In the first case, the developer would have to, on selection definition import, search through the DOM and identify all the entities they want to enable this selection mode for. Then, their resulting definition would be quite simple: the selection condition would check if there is a mousedown event on one of the identified entities. Then when the mouse is released, the onSelectionEnd callback check to make sure its target was the same as the triggering mousedown event, and if so it would perform the appropriate saving procedure.

In the case of the Click and Cross cursor, the developer would have to manage the area cursor and define the available selection targets on selection definition import. The Siphon selection condition would then check if there was a mouseup event on a supported target, that the mode isn't already active (users click again to cancel Click and Cross), and that the user hasn't "crossed over" to select one of the elements. The onSelectionStart callback would setup the click and cross interface, the onSelectionChange would draw the line from mouseUp position to the current mouse position, and onSelectionEnd would clean up the interface, and if the user crossed a selection arc, select that element. These custom selection interfaces, along with some of the build-in Siphon selection definitions, could all be made available to end users as ways to select content on a webpage. Once the user completes a selection using one of these tools, the developer could then create a Siphon annotation object, using the bounding box created by the force touch selection or the DOM nodes from the Click and Cross and Entity Quick Click selection modes, to allow for a rich view of the content selection, or generate a persistent reference to the content on the page.

5.4.2 Richer Reuse and Consumption

We also examine how Siphon can simplify and augment tools designed for further organization and action on content from pages. One of the more modern sensemaking tools, IdeaMache [98], allows users to collect rich web content (images, google maps, video, audio, etc.), organize that information on a free form canvas, add additional annotations to the canvas, and then use this to drive a presentation. IdeaMache supports collecting content mainly through drag and drop and copy and paste interactions.

Alternatively, IdeaMache could leverage Siphon in a browser extension to allow for

even more extensive set of supported content: highlights from web pages with embedded links, styled, interactive HTML snippets from web pages, and video and audio from a variety of sources, instead of just a few providers. Another core component of IdeaMache is the ability to revisit the source for a piece of content. Instead of just going to the page for that content, using Siphon, IdeaMache could direct users to the exact point in a web page the content came from, and even highlight or draw a box around the content. By using Siphon, IdeaMache's developers could focus more on the difficulties of organizing, displaying and managing web content on a free form canvas, rather than worrying about supporting the selection and extraction of web content.

Another system, Nota Bene [157], provides students with a website to collaboratively annotate PDF documents. Users can select content, and then leave questions or have discussions about the content in a sidebar (or the "margins") of the document. Using Siphon, Nota Bene could easily be adapted to work with general web content. Using the provided selection tools of Siphon, students can easily reference content on a page. The annotation objects produced by Siphon can then be augmented to include comments as a property, so individuals can have discussions about certain pieces of content. Because Siphon persists the original DOM location on the creation of an annotation object, other users who visit the page can have the annotations surfaced on their view along with any associated discussion. Thus the main implementation thrusts for making a version of Note Bene for web content would center around the design of the sidebar interface, the popup annotation editor, and any discussion features the interface needs to support. This shunts the technical overhead of being able to create persistent references to content on the web from the developers to Siphon, allowing them to focus on the challenges unique to collaborative learning with annotations.

5.5 Discussion

By recognizing this core annotation workflow of selection, extraction and consumption, Siphon can enable developers and researchers to create more powerful web annotation tools with a simplified Javascript API. Beyond supporting annotation, Siphon has the potential to augment other online activities, such as discussion or collaboration. Many news articles articles now contain embedded discussion forms for users to provide their personal opinions or share additional information about the topic. While these tools offer the ability to respond to other comments, they either lack support for responding to the original content, or only provide very basic support referencing that content (i.e. only highlights). A developer, wishing for richer discussion around the article content, could implement Siphon as a way to allow for inline discussions, similar to Note Bene, or as a way in a comments thread, to reference the article content with a variety of selection tools.

Because Siphon is implemented as a vanilla Javascript library, it can be used wherever a web context is present, this includes Electron application on a desktop, mobile progressive web applications, or browser extensions. At the current moment, this does limit it to only web content, for example it would not work with PDFs or eBooks. However, because of the current strong push to use web technologies for application development, there are a number of libraries, such as PDF.js, that take these documents

and render them into a web context, where Siphon could then be extended to provide annotation support. Going a step further, there is the potential to adapt it to different Javascript rendering environments, such as React Native, for support on other platforms, such as mobile devices or even VR.

Aside from platform limitations, Siphon, through it's architecture, has several boundaries for what annotation support it provides. For example, as noted in the Click and Cross example, it would be up to the developer to implement the bubble selection cursor. Siphon generally assumes that all DOM elements are valid targets for selection and extraction, all of which are nested in a large hierarchy. The bubble cursor, on the other hand, requires a set of distinct targets for selection, therefore the developer needs to specify those and then provide those as the outcome to Siphon, rather than relying on Siphon's built in tools for graphical DOM element selection. Like any toolkit, Siphon has the potential to introduce whorfian effects into the development of future annotation selection and extraction techniques. Because Siphon is an add-on library, rather than a standalone interface / language for annotation design, developers and researchers should be able to work around any limitations by falling back to the lower-level Javascript APIs.

Lastly, Siphon's graphical DOM selection and HTML snippet generation algorithms, while fairly robust, are heuristic algorithms depending on how a developer designed their webpage. For example, pages that use extensive Javascript for layout computations might have a number of runtime computed styles that appear "fixed". Therefore, the HTML snippet generated by Siphon might appear to be malformed, depending on the content selected. While Siphon cannot cover all edge cases, one possible alternative is to take a screen shot of the content and then allow the user to toggle between that and the interactive HTML snippet. Ideally, as CSS tooling continues to expand to meet the layout needs of web developers, there will be less and less dependence on alternative techniques for layout (i.e. Javascript) and Siphon can continue to grow to support annotation more reliably.

Chapter 6

Distil: Categorizing on the Fly

With tools built that allow users to find and triage information efficiently, I then focused on assisting users with generating a model. Because Sensemaking is a cyclical process, often with fairly quick iteration, a user's model, data sources, and triaged information change rapidly. Manual or cluster-based organizations might work for one instance of the cycle, but could become quickly outdated the next, requiring significant effort on the user's part to reorganize what they've collected. I developed Distil to assist users with this model generation challenge: through interactive "smart categories", users can define auto-updating categorizations that automatically pull in relevant existing and new information. These smart categories allow users to more efficiently perform the processes of classification and schema induction on their collected data through streamlined categorization. With Distil, users were able to quickly create and adjust their categorizations, using them to both more deeply explore the dataset as well as organize it.

6.1 Introduction

The incremental, mutating nature of the structure of a user's mental model poses challenges for existing tools that aim to support sensemaking. For example, one major class of sensemaking tools focus on providing tools for users to easily manually create structured representations of collected information, either through tags (i.e. Hearst's triage tool [59]), categories and clusters (Clipper [79, 80]), or spatial arrangements (i.e. Sandbox [154], IdeaMache [98]). While these tools provide end-users with fine-grained control over the document and information saved in their final output, doing so requires a large amount of work to manually create an organization. Furthermore, as users encounter new information and their understanding of the space grows, they have to apply a significant amount of effort to adjust their workspace to match their new understanding or add their newly encountered content.

Alternatively, a significant amount of work has gone into building data exploration interfaces, where sensemakers can utilize a system-generated structure through facets [58, 87, 90] or clustering [31, 32, 121, 156] to explore a given dataset. These techniques allow users to quickly sift through a large set of data while also learning about its structure [58, 138]. However, these techniques often assume a fixed dataset with all

the possible information already present, whereas in the sensemaking scenarios we are interested in [130], the set of documents a user considers grows and changes as a user continues to serially encounter new content areas and topics they should pay attention to. Therefore, a user would either have to wait to cluster until all information is collected, or recluster their information and risk ending up with a different automatic structure. Additionally, the relevant portions of documents clipped by users are often short, sparse, and can belong to multiple different categories, raising additional challenges for computational approaches.

Traditional categories, therefore, are not a good match for the sensemaking process: they require users to manually assign items to them, don't automatically incorporate new information, and are generally fixed and difficult to change without significant manual effort. To tackle these challenges, we introduce a new sensemaking support tool, Distil. Distil allows users to continually refine the structure of their digital workspace through user-generated, keyword driven, streaming categories. A *streaming category* automatically pulls in relevant portions of the information a user has collected based on a set of user-specified category keywords. This not only has the potential of allowing users to create categories more efficiently, but also allows the system to automatically assign new information to existing categories as users explore the web and save more information. These categories also allow for iterative evolution of a user's sensemaking structure. While a user builds up their understanding of the information space, they can quickly add, refine, or remove their streaming categories, seeing updates in real-time. As this evolution continues, they can further refine the snippets of information automatically pulled in by the system, and interleave their personal thoughts and comments with the automatically categorized content.

The key contribution of streaming categories is that they enable *anytime sensemaking*. Instead of having obsolete structures created at the beginning of their process because they didn't know the space [80] or having to wait until the end of their process to organize anything, they can create a streaming category when they encounter information that is relevant to their goals. This category can be refined at the same time it is being used to understand and organize collected content. To evaluate how users utilized streaming categories to assist their sensemaking process, we ran a two-phase user study in which we examined how participants' structure evolved over time and helped them organize new information.

6.2 Related Work

Many tools for assisting users during the exploratory search process do so by providing up-front categories through clustering [31, 32, 39, 85, 121, 156] and facets [87, 88, 138]. Apolo [32] takes a unique approach to this by having users build up a categorization through exemplars rather than providing a top-down set of categories produced by an unsupervised algorithm. This allows for the user to iteratively build up and adjust their information landscape as they continue to learn. However, Apolo is designed to work with a metadata-rich data source, while general web-based sensemaking often occurs with small, sparse partial sections of a document, making it difficult to provide a mixed-initiative system like Apolo with enough useful data for clustering. Grouper

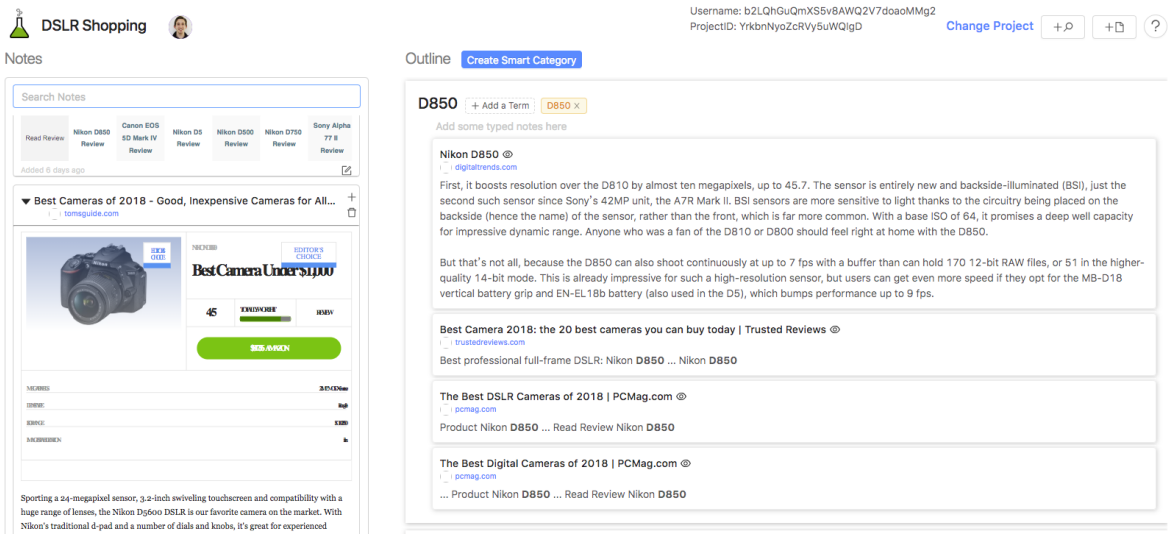


Figure 6.1: Distil's interface: The left shows the full notes a user has for this task, while the right is a free-form text editor with streaming categories

[156], alternatively, works on the subdocument level by clustering document snippets returned from a search. While these categories can be extremely beneficial for getting an initial overview of the space and inform users what features they should focus on [90, 138], they often do not align well with user's final mental model from their sensemaking process [80] and can become obsolete as the set of information gathered continues to grow. Therefore, while automated clustering techniques are very beneficial for gaining an overview of a fixed information space, they aren't adaptable enough for the sensemaking process where the information space and the user's mental landscape are constantly updating.

Due to the limitations of the above approaches, other researchers have focused on tools that introduce helpful metaphors or specific organizational layouts that assist users with collecting and organizing information during different phases of the sensemaking process. These include early stage tools for helping users manage and re-find sources, such as WebBook and WebForager [24] which utilize a book metaphor for managing pages, Elastic Windows [72], and Webcutter [101] which presents URL collections in a tree, star and fisheye. These tools provide significant support for managing sources, but are generally limited to entire documents and are focused on supporting retrieval of existing information rather than generating organizations for explanation. Alternatively, organizational tools attempt to provide flexible ways to distill and group content, such as Clipper [79, 80] which allows users to assign attributes to a clip at the same time it is being saved from a web page, IdeaMache [98] which uses a spatial layout for helping users group and manage content, and Hearst et al.'s triage tool [59] which uses a streamlined interaction for categorizing and tagging documents. While manual organization tools are extremely beneficial for creating detailed, easily interpretable artifacts, they can require a significant time and effort investment by the end user to create, and can become obsolete during the sensemaking process as a user's understanding of the space changes [80].

These existing tools support individual phases of the sensemaking process [123, 130] in isolation, but this may diverge from how information foragers actually make sense of different domains; for example, it is common for additional foraging to occur after initial information structures are created. This implies that by focusing on the separation between sensemaking phases, support from existing tools for organizing information leads to a considerable amount of post hoc efforts to properly update the information structures created earlier in the sensemaking process. Our approach aims to complement this commonly found perspective in organizational support, by considering the cyclic nature of the sensemaking process. We ask, "how might we support fluid movements between phases of the sensemaking process with a system for organizing information while minimizing overhead from its users?". To answer this question, we propose a novel concept of "streaming categories" to support anytime sensemaking in an interactive, Web-based prototypical system **Distil**. We take inspiration from IntentStreams' [9] query refinement interface – their constantly updating streams in response to user feedback allow users to iteratively adjust their search. At the core, Distil supports iterative refinement of information foragers' sensemaking workspace. We observe that this operation of dynamic refinement affords fluid movements between different sensemaking phases. In the following sections, we describe how Distil is designed to achieve this goal.

6.3 System Description

Distil is a prototype system for organizing clips of information users have collected from the web. The core component of Distil is the streaming category: a keyword-driven filter that continuously categorizes new content added during the sensemaking process. Categories can be further refined and explored through a few different mechanisms: adding or removing filter terms, viewing the original source content for the snippets pulled into a streaming category, adding additional clips to a category, and specifying more specific summary text for a clip pulled into a category. Streaming categories allow users to begin structuring their data at any point in the sensemaking process, without having to worry about the trade-off between structuring too early (thus creating obsolete structures) or too late (having to deal with an overwhelming amount of content). In order to describe how streaming categories are supported in Distil, we first introduce the concept of *clip* and *outliner*.

6.3.1 Clips

Distil's streaming categories are designed to work with "clips" from web pages – a subset of content selected from a page by a user. Distil works with several types of content, ranging from simple highlights to entire HTML pages. The three main content types supported are simple textual clips, HTML fragments of a page, and full HTML pages. The style of HTML fragments and pages are retained from their original source, and are rendered in a fully interactive manner (Figure 6.1, left). All of the clips a user has collected are shown in still in a list on the left-hand side of the interface.

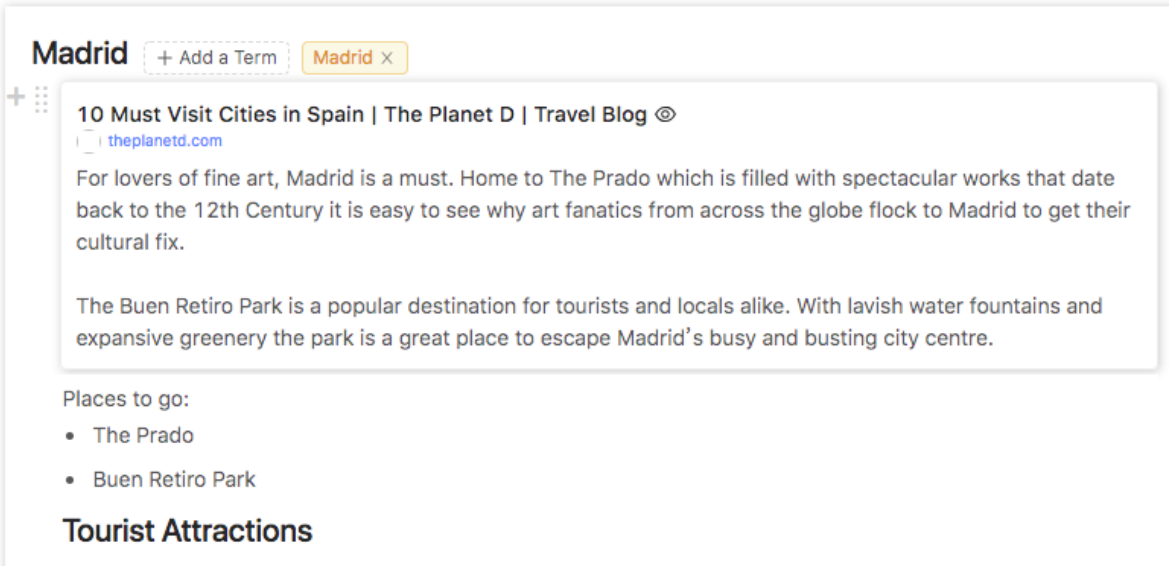


Figure 6.2: The Distil outline interface – shown is a streaming category with both a snippet inside of it as well as manually entered user text

6.3.2 Outliner

Distil allows users to create hierarchical outlines of clips and manual notes using a free-form text interface, allowing a user to interleave their personal thoughts and opinions with clips in a category (see fig. 6.2). In this interface, clips and sections of text are treated as individual, nestable, reorderable elements. As a user continues to refine their outline, they can seamlessly add, reorder, and remove these chunks of content, similar to a popular note-taking application Notion [142]. This document is placed in the right portion of the Distil interface (fig. 6.1, right).

6.3.3 Streaming Categories

To streamline the process of adding and organizing clips in their outline, a user can create a streaming category. A streaming category represents a particular topic within the data a user has collected that they would like to explore further; retain for later use such as comparison or sharing; or as a way to remind themself and surface important content. For example, consider a scenario of a user trying to find good restaurants in a city they are visiting. Initially, they may go through several listicles to find popular restaurants, as well as options listed on restaurant-rating applications such as Yelp around the area they are staying in.

Using some prior knowledge about different types of cuisine, the user then starts with making a few different streaming categories: a 'Japanese' one, an 'Italian' one, and a 'Moroccan' one. As these categories are created, the category names are used as the initial search term for gathering clips that were saved in the system, such as reviews extracted from Yelp of different listicles. Distil uses the standard Okapi BM25 ranking function for ordering the snippets [127]. Long clips in a streaming category are summarized around the matched keyword, similar to a search snippet. Users can

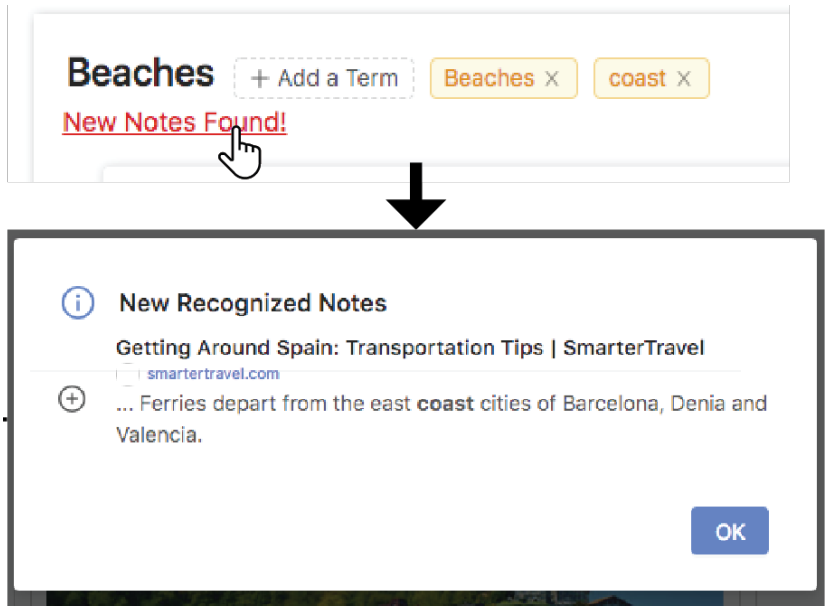


Figure 6.3: When a new note recognizes a streaming category, a small notification appears. Clicking on it opens a modal where users can then add that note to their streaming category

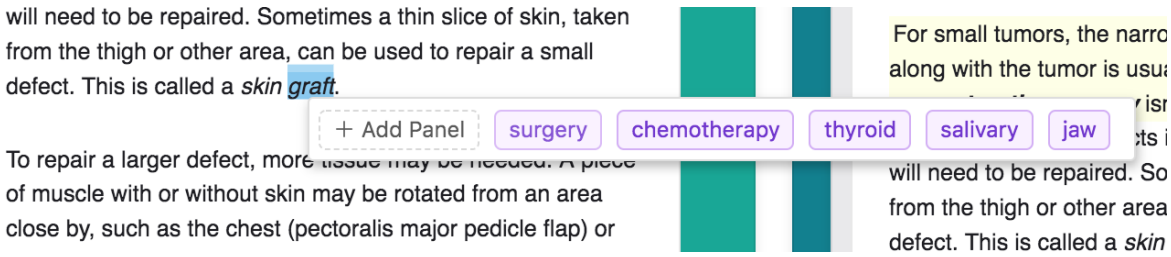


Figure 6.4: Toolbar for adding a term to a category

rearrange these snippets, delete irrelevant ones, and update their summaries to be more concise. Additionally, a user can revisit (focus on) the full clip where a snippet was extracted from, in order to gain more context or explore additional details not present in the summary.

Noticing there are not a lot of clips in the Japanese category, they focus on expanding their search for more potentially good Japanese restaurants. As they continue to collect clips, Distil continually checks the content of these clips against their current streaming categories. Any streaming categories matching that new clip will display a small notification under them saying "New Notes Found!" in red. So, in this case as the user continues to collect information about Japanese restaurants, their Japanese category will display a notification indicating new content. A user can then click on this and add any of the desired matching clip snippets to their streaming category (see fig. 6.3).

As the user continues to learn more about the restaurants, they might realize that the initial keyword Japanese is not capturing some of their relevant clips that did not explicitly mention it. To resolve this, they add more keywords as an additional filters

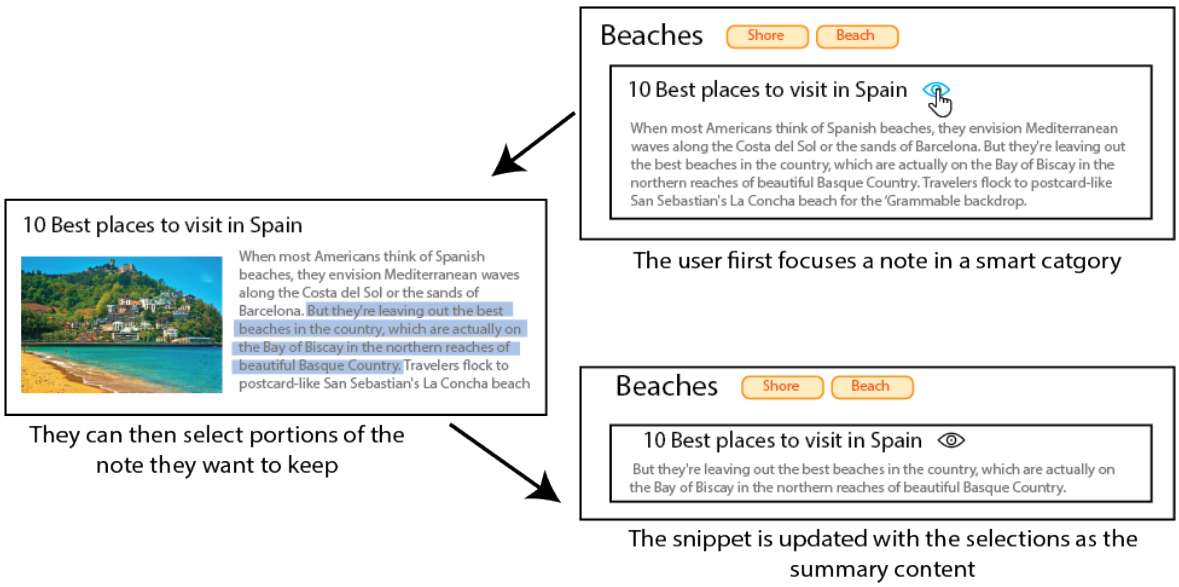


Figure 6.5: The interaction for editing a note summary

to their Japanese category, such as "ramen", "sushi", and "takoyaki". The union of the results from these filters is then computed, and if anything new matches, they will get a familiar "New Notes Found!" notification under the category where they can add any desired missing snippets. Users can add additional filter phrases in a few different ways: they can enter them directly, they can choose a phrase in the text of their clips to add a term from (see fig 6.4), or they can use the search bar at the top of the clips section to add a phrase. When using the search bar, they can preview what will appear in the resulting category before adding it. If adding a phrase from the text, we enhance the process by suggesting the top 5 existing categories that them phrase would most likely match by computing similarity of the word selected against each category name using GloVe [122] and then rank the results.

As they read through the clips in the Japanese category, the user notices that the summary text is not very useful, and somewhat verbose. In response, they modify the snippet text by selecting new text from the clip to show there instead (see fig. 6.5). They can also choose to show the entire clip instead of just the snippet, which can be useful in the cases that an entire clip is relevant, or it contains some highly graphical data (like a table) that would be better viewed in its original format. Finally, in the space at the top of the Japanese category, the user manually jots down the top restaurants they want to go to based on their category's data.

6.4 Implementation

Distil is implemented as a web application. The application is written in Javascript using Facebook's React library for interface rendering, and Google's Firebase database service for data persistence. Lastly, it uses the open source Lunr.js search engine to provide the filtering capabilities of the streaming categories.

6.5 Evaluation

We evaluated whether Distil effectively supports organization of information and its gradual refinement during sensemaking through a controlled experiment. In a within-subjects study, participants were asked to complete two different sensemaking tasks, one with the support of Distil's streaming categories and one without. The order of tasks and conditions were randomized and counterbalanced.

We recruited 17 participants (9 identified as female) from a local participant pool. The average age for the participants was 26 ($\sigma = 8.73$). Nine participants reported as college graduates and 8 of them are currently enrolled in an undergraduate program. The study took approximately 90 minutes and each participant was paid 15 USD. The study consisted of 6 different sections: a demographic pre-survey, a 7-question Maximizer-Satisficer Questionnaire [116], a short interface training, two sensemaking tasks, and finally a post-survey asking their experience. We utilized the Maximizer-Satisficer scale to estimate how much effort a participant might spend in a sensemaking task. During the training, participants first watched a short video explaining the system, and completed a simple task on researching a DSLR camera to recommend to a friend.

For the main tasks of the experiment, participants were given the following prompts:

- **Spain Scenario**: In this task you will organize / summarize information for planning a trip to Spain with a friend or significant other. You need to consider what areas you'll go to, what attractions you might visit, and how you'll get around.

- **Diabetes Scenario**: In this task you will organize / summarize information for a friend / significant other / child around the treatment, side-effects, and long term considerations of managing Type 1 Diabetes. They were just recently diagnosed, and are looking for some input in how to manage their condition.

In each of the tasks, participants were given a predefined set of clips, taken from the top 10 Google results for each of the scenarios. These were extracted HTML fragments of the different sections of those pages. The Spain scenario had a total of 71 notes, and the Diabetes task had a total of 41 notes. Initially, users were given a random sample of $\frac{2}{3}$rds of these notes and asked to spend approximately 20 minutes organizing them.

Next, participants were given the remaining $\frac{1}{3}$rd of the notes. They were asked to spend the next 10 minutes incorporating them into their existing summary, or change their structure to incorporate the notes. This was intended to simulate finding additional information during sensemaking, since users were not collecting clips themselfs. Finally, participants were asked what they would change if they had more time to organize the information, and finally given 5 additional minutes to make any changes.

After finishing the two tasks, participants completed a post survey, which elicited feedback comparing the base condition to Distil, and asking their opinions about the different features of Distil. Throughout the user study, we logged the different activities and behaviors of users in the interface.

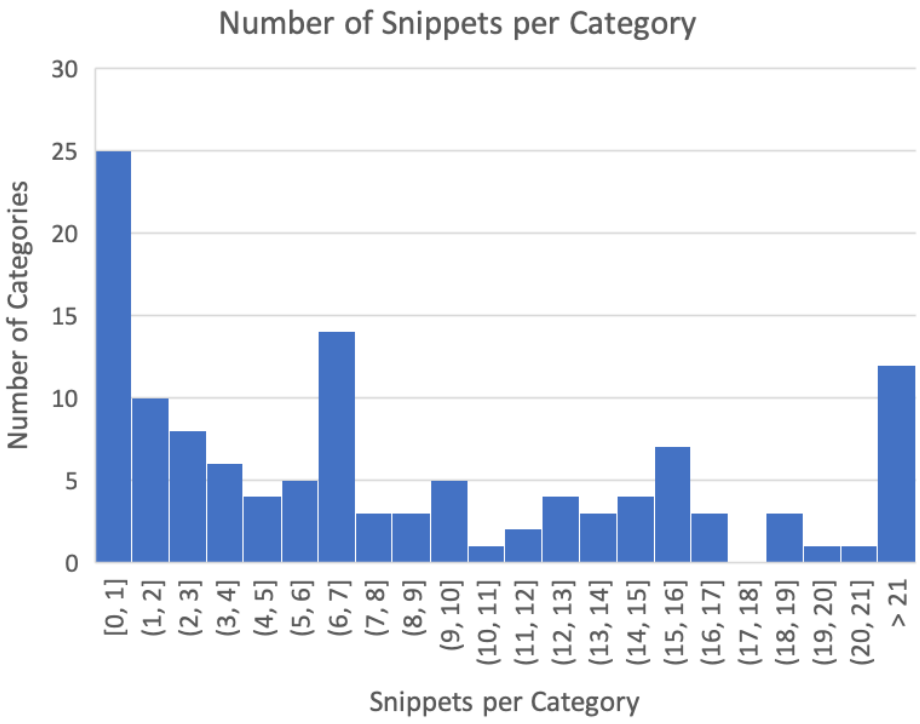


Figure 6.6: A histogram of the number of clips in the streaming categories

6.6 Results

As participants utilized Distil, we collected data on the usage of the different features of the Distil tool. In the post-survey, we collected qualitative data through 8 self-report Likert scale questions (1-5) asking them to compare the control condition to the complete version of Distil. These were analyzed using a mixed effects model with the scenario of the task (Spain vs Diabetes), the type of interface (Control vs Complete), responses to a Maximizer-Satisficer Questionnaire [116], and their interaction as predictors. Additionally, we collected several free-response questions to gain deeper insight into users' experience with Distil. Using this data, we explore:

- How users built organizations using the streaming categories
- How users were adjusting their organizations over time
- How users responded to new information being added to their workspace
- If the streaming categories were transparent in their operation and let users feel that they were in control of their data

6.6.1 Supporting Organization

First, in order to get a sense of how users were using streaming categories to organize their data, we looked at the Distil system event data. We saw users utilizing the streaming categories fairly significantly: on average each participant created 11 streaming categories

Type	Event Name	Percentage of all Category Events
Adding / Removing	createCategory	*
	deleteCategory	3.89%
Evolving	addFilter	14.75%
	removeFilter	7.24%
	changeCategoryName	2.95%
Exploring	focusNote	29.36%
Curating	deleteSnippet	32.17%
	saveFullNote	1.07%
	saveSelection	8.58%

Table 6.1: The breakdown of streaming category event frequency * createCategory is listed for easier future reference

($\sigma = 5.29$) with each containing approximately 10 ($\sigma = 12.55$) clip snippets at the end of the study (See Figure 6.6). Additionally, users manually removed around 2 snippets ($\bar{x} = 1.88, \sigma = 5.22$) from the category during the study. These results suggest that users were able to actively use the streaming categories as a way to organize their data during the study.

To dive deeper into whether participants were creating category for exploration or as a means to actively build out a data categorization, we looked at the events performed on individual categories. After creating a category, users continued to perform on average 4 additional actions ($\sigma = 4.86$). These actions could include adding an additional filter (addFilter), changing the category name (changeCategoryName), focusing on a clip added to the category (focusNote), removing a filter (removeFilter), manually adding a clip not automatically captured by a category to it (saveFullNote), or changing the default clip snippets (saveSelection) (See Table 6.1). About one third of the time users were exploring the data pulled in by the category, while the other two thirds of the time, they were further refining the information captured by the streaming category. This split in activity suggests that users wanted to further verify that their streaming categories were working appropriately (focusNote), while also tweaking the final output of the category so that the persisted information was more useful (addFilter, removeFilter, saveSelection, deleteSnippet).

Unexpectedly, there was a bimodal distribution to the number (See Figure 6.7) of actions performed on each category, where a large number of categories were either considerably unchanged after creation or frequently refined and evolved by the users. This can be attributed to the way categories appeared to be used for exploring potential structures – on average users delete about 3 ($\sigma = 3.08$) categories and when a category was deleted, this typically occurred within the first two actions ($\bar{x} = 2.23, \sigma = 1.52$). We noticed a similar trend for filters: Users added approximately 9 additional filter phrases ($\sigma = 7.24$) to their streaming categories and approximately half of the time ($\bar{x} = 5.22, \sigma = 4.90$) they removed one of those filters. There was a high level of variation in both of these situations, suggesting that some users could be experimenting with different structures or phrases for their structures more so than others.

At the end of the study, participants compared Distil to the baseline system with

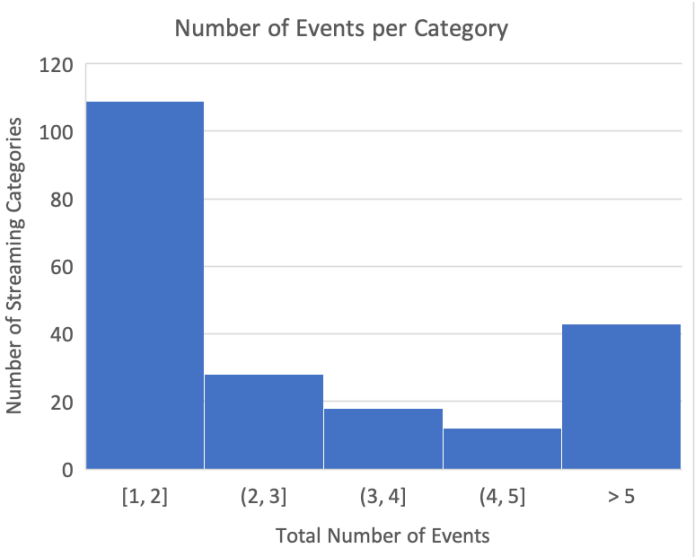


Figure 6.7: A histogram of the number of total actions performed on the streaming categories

a series of qualitative Likert and free-response questions. They reported that it was significantly easier in Distil to categorize the first set of information ($t(16) = 7.29, p < .01$), as well as the added, subsequent information ($t(16) = 8.23, p < .01$). When further probed on whether Distil helped them categorize their information, users cited the benefits of: "It was visually appealing to have a lot of data/information organized", "easier and saves labor" and "could quickly decide what info goes where". We also asked users in what situations they would imagine these streaming categories being useful – one specifically wanted it for travel planning – "I like the idea of using this system to form an itinerary since it can pull from multiple sources and compile information into one document" – while another wanted it for the personal notes they've taken: "It would be useful to be able to quickly categorize and summarize personal notes." Others also mentioned using it for a job search, academic literature review, meal planning, wedding planning, and shopping.

6.6.2 Continuous Adjustment

In addition to proving categorization support, Distil was designed to allow users to easily, flexibly and transparently adjust their digital workspace to match their current understanding of the information space they are learning about. Throughout the entire time period with the Distil tool, participants (See Figure 6.8) continued to add, modify and remove streaming categories. While there was some initial front-loading of categories, users were modifying and updating their structure in Distil as they learned more about the information space.

In the second part of the study, participants had the remaining one third of the clips added to their workspace, as a way to simulate gathering additional data during sensemaking. After the additional content was added to their task, we continued to see this gradual refinement (See Figure 6.9), albeit at a much lower intensity than initially.

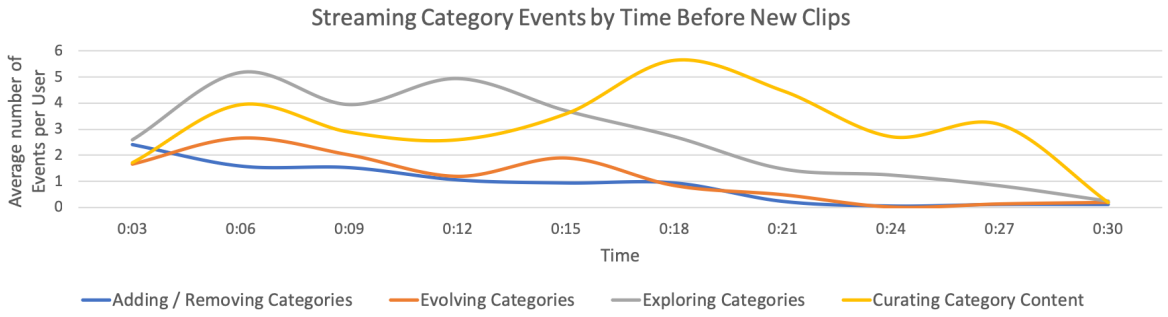


Figure 6.8: The average number of each type of event by time for a streaming category before new clips were added

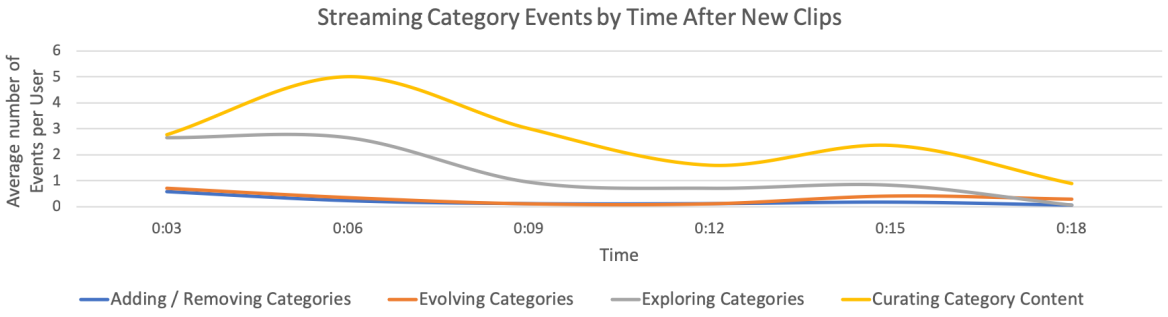


Figure 6.9: The average number of each type of event by time for a streaming category after new clips were added

We didn't notice any significant changes to users' structures, which was in line with our expectations for how streaming categories should allow for gradual refinement and incorporation of new data. The majority of the activity was in exploring categories, as individuals most likely wanted to examine and verify the new data was correctly pulled in, and in curating category content, as participants continued to filter out and update some clip summaries.

To further confirm that users were using the streaming categories as a means of incorporating new data and making slight changes rather than just performing a post-hoc categorization, we looked at the qualitative data. When asked about how they refined their structures in response to new information, users mentioned mainly small organizational or filtering adjustments, rather than any drastic change to their structure. Only 5 participants in the Distil condition and 3 in the simple condition answered yes to the question: "Did your structure from the first phase change?" In our qualitative survey, we also didn't find a significant difference between the Distil and base condition for the question "How easy was it to change my organization in each interface?" (Distil: $\bar{x} = 3.17$, Simple: $\bar{x} = 2.88$). We hypothesize that this was due to individuals not really having to ever truly perform a drastic change to their organization; rather they were just refining and adding to it as they went. We saw additional evidence of this in the free text responses as well:

> "I mainly changed the sequence in which the notes appeared. Also I added headings to specify the spots which are closer to Barcelona and Madrid."

> "I ended up moving the life factors to their respective category, which then left the insulin category all by itself, which aids in clarity."

> "With the new notes, I split one larger treatment category into 2 smaller categories. I added a new category as a result, and removed extra notes from the broader category."

> "I maintained the general structure but partitioned new information in respective categories."

6.6.3 Maintaining Control

Lastly, Distil takes a fairly simple, keyword-based approach for creating and managing streaming categories out of concern for transparency and user control. In the self-report questionnaire, participants agreed with the statements "I found it useful to create streaming categories and have them automatically pull in snippets from notes" ($\bar{x} = 1.76, \sigma = 0.97$) and "I found it easy to create and manage my streaming categories" ($\bar{x} = 1.88, \sigma = 0.99$). While these suggest that the interface was fairly intuitive and provided sufficient explanation for users, Distil was still overwhelming for some users –

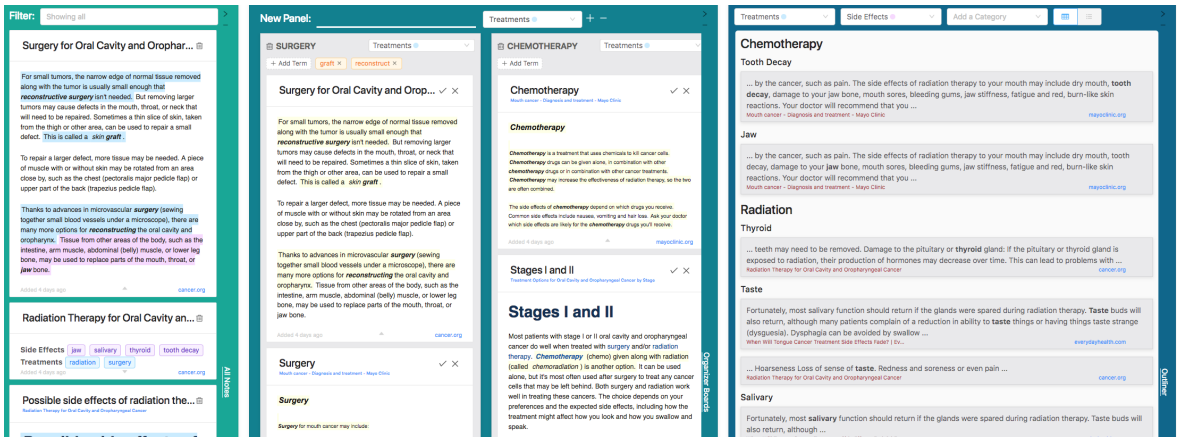


Figure 6.10: Original version of Distil

one specifically stated: "I wish I had more control over what was in my notes". Others mentioned it being a bit too greedy / automatic, which ended up requiring pruning: "As a concept it could be useful. I hate 'pruning'. Especially having to prune through data that I didn't choose." "Streaming Category places every remotely relevant link in the category, some of which is repetitive." Going forward, future interfaces for supporting intelligent categorization during the sensemaking process need to provide extremely transparent feedback about the operations they are performing and how they are organizing data.

6.7 Discussion

The combination of user behavior along with the qualitative responses from participants suggests that Distil is able to effectively support "anytime sensemaking" by allowing users to create categorizations that quickly and efficiently pull in data, allow for continuous adjustment, and support transparency in their operation. Participants leveraged streaming categories as a way to both further explore the dataset provided to them in the study, while also adjusting its filters and summaries to produce from their perspective, useful categorizations of the data. These categories were effective at pulling in new information when it was provided to users, and only required minor tweaking to remain relevant and useful. Lastly, the categories seemed to be mostly understandable in their operations, however for some they were still too automatic.

6.7.1 Earlier Versions

In order to achieve the above results, Distil went through several different iterations, with each version providing additional insight into what users wanted from a dynamic sensemaking interface. The original version of Distil featured a layout similar to Trello, where users put their notes into categories instead of summarizing them. This then separately fed into an outliner view. Users, however, found it cumbersome to first have to put their notes into a panel and then manage them in the outliner. Therefore, we

obviated the second stage, and had users organize the information directly into their final outline.

Additionally, the original version had no easy method for users to record their thoughts or opinions about a category. In another intermediate version of Distil, we added a comments section at the top of the organization cards, however we discovered that users were just copying and pasting text from the original notes into those sections because they had no way to associate their notes with a specific piece of text. This led us to the final, more free-form design where users could easily interleave manual notes with the snippets automatically or by manually extracting from their notes.

6.7.2 Alternative Representations

In the tasks for the user study, we picked two fairly common sensemaking task domains: medical and trip planning. However, through the feedback from our participants in both of our studies, we discovered a few other potentially promising domains for Distil. These include learning a new academic subject, performing research, wedding planning, managing personal information scraps, literature review, performing a job search and meal planning. While Distil was built and designed to be a general purpose sensemaking tool, there could be data from these specific domains (such as a medical database, Amazon shopping results, ability to work with PDFs) that could bolster its application to other situations.

Additionally, users might not want a document outline as their final information form. As we discovered in our trip planning scenario, a number of users wanted a map. As noted in our preliminary study, tables were extremely popular for shopping tasks, as participants had a consistent set of criteria they were comparing each item against. Additionally, one participant in our study event stated that "would have liked some pictures, data, or charts perhaps." While this initial version of Distil did not feature those data forms, combining the features of Distil with data visualization tools could provide significant benefit to users. This would add another transformation layer on top of the streaming categories, possibly leading to a very powerful tool for individuals working with unstructured textual data.

6.8 Limitations and Future Work

While Distil is designed as a general purpose sensemaking tool, there are some situations where it provides limited support. Due to its keyword-oriented design, it could be overly greedy in pulling in information, adding irrelevant clips to a category. Additionally, there could be ambiguous phrases across clips, for example one clip could be talking about Jaguar – a model of car – while another could be referencing the animal. Three possible solutions include breaking up larger clips into smaller, more focused ones, using more advanced text processing and searching (e.g., extending the word vector approach we use for suggesting the category a phrase should be added to), or using named entity recognition techniques to provide better filtering functions.

Distil does use some basic information retrieval techniques to support quick organization, there are a number of opportunities to further expand on these to both

decrease the user's mental effort, and ensure summarized information is comprehensive and representative of the data collected. One approach would be automatic category generation - using an unsupervised clustering algorithm and word embeddings, users could have categories suggested to them based on the data that has been collected. Another could be more structured entity extraction and presentation where Distil could pull out specific entities, such as phone number, addresses, or headings, and utilize those as components for creating even more structured organizations [19, 47]. These would be especially useful in the later stages of the sensemaking process, where a user will more likely want specifics about the topics and items they've focused in on.

One possible danger of this approach is it could make users only perform surface level evaluations of their collected information, rather than considering it more deeply. One of our participants directly pointed this out: "It was definitely very helpful but I thought it was easier to neglect some details. In the plain text, in order to filter/categorize well, I needed to really read each note. In the advanced system, I just had to type the category and it would automatically categorize information (useful and not useful information)." This might have been due to the artificial situation imposed by the study: users were categorizing information they didn't directly clip, so they were also doing a lot more discovery than a traditional sensemaking scenario. We chose this study design due to the time and mental load concerns from performing a full-fledged sensemaking task, as well as concerns that different users would collect vastly different amounts of content depending on how much they cared about the provided topic. A larger deployment study would help to provide a clearer picture about how much the streaming categories are used for additional exploration versus pure categorization.

Looking forward, Distil is just a first step at supporting the broader concept of anytime sensemaking. The streaming categories offer a unique capability, in that users can use them for both proactive planning – knowing they are going to come across beaches on a trip to Spain, so they proactively create a beach category – and retroactive retrieval – they just discovered that traveling by train is the best way to go, so they create a train category and see how many destinations talk about trains. This also happens at a higher level in many sensemaking tasks: a user who is planning a vacation knows it's going to be a complicated process, so they might front load some of their organizational efforts based on their past experiences. Conversely, someone researching a humidifier might expect there to be only a few options that they might choose from; however after learning of different types, brands, models, etc. of humidifier the user might recognize that they need help keeping track of all of this information. Developing techniques that are able to flexibly respond to changes in a user's research trajectory could help to produce tools that are more readily adopted by end users, as they could exist as lightweight collection and organization tools and then morph as a user's needs grow. Some possible instantiations of this would be tools that allow users to transition between structures (like convert a simple outline into a table), find additional information about an entity or topic from other web pages that they've already visited or are planning on visiting, or providing categories a user should explore / consider based on the structures produced by other individuals performing the same task.

Chapter 7

Proposed Work

In my previous work, I developed systems that support an isolated portion of the sensemaking process. Bento was designed to examine supporting the portion going from unknowns, to information sources, to the trove. By creating a workspace for sensemaking tasks, Bento was able to provide users with a way to queue up things to work on, highlight important sources and remove irrelevant ones, and begin to backfill a mental structure they could use for easily resuming their tasks. Siphon was built to more closely model and support the processes occurring between the sources and information trove – elimination and semantic fit. Due to changing levels of uncertainty throughout the process, the amount of elimination and the ability for individuals to judge semantic fit can vary greatly. By providing tools for both early stage (large boundaries and context) and late stage (specific, pointed extraction), Siphon enables users to capture the right level and context of information, while supporting a connection back to the original content. Finally, Distil supports the flow of data from the trove into the model by supporting schema induction and classification. Distil through the use of smart categories allows users to create and maintain a categorization scheme without having to manually organize or refactor their content.

While these prototypes have allowed me to explore and develop tools for supporting some of the core portions of the dataflow-centric sensemaking modal, there are still several missing mechanisms and routes. Additionally, all of these tools currently exist as independent prototypes, which have only been tested through lab and field experiments. Ideally, to properly support this entire process, these tools would exist in harmony and be capable of passing off data from one to the next as the user moves through this process. Therefore, I propose the following:

- The integration of the three previous systems (Bento, Siphon, and Distil) into a cohesive, deployable system that would be able to test the effectiveness of dataflow-centric sensemaking tools
- The development of an additional component that would be included in the aforementioned system that would support the remaining arc from model to residuals, that could then be utilized by the end user to use the system as an end-to-end solution.
- The deployment of the system to a large (1000+) set of users to develop a

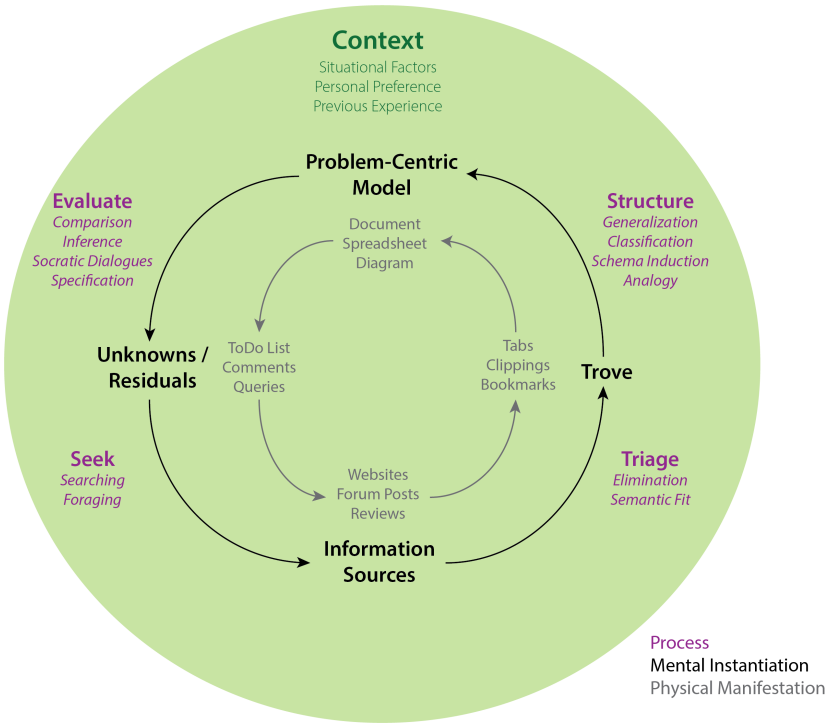


Figure 7.1: A dataflow-centric sensemaking workflow model

cohesive dataset of sensemaking that could be leveraged for further analysis of the sensemaking process and drive computational support.

7.1 Fuse: Integrating the parts

In order to take advantage of dataflow-centric sensemaking, I've begun the development of the Fuse system along with my close collaborator Joseph Chang. Fuse takes some of the core design elements of Bento, Siphon, Distil, and another system Fusion, to create an in-browser sensemaking workspace. Fuse, taking inspiration from the Fusion system, exists as a sidebar that can be opened on any web page. From this sidebar, users can create and manage their various ongoing sensemaking tasks, which exists as tabs, by collecting webpages and any content on these pages as a "card". These cards can be structured into an outline format, put in a folder, or combined into a single card.

As of now, Fuse mainly embodies the concepts from Bento and Siphon. Users can manage different sensemaking processes, record their activities in an in-situ workspace, and appropriately triage things they save into their workspace through the use of categories or colors (Figure 7.2). Fuse utilizes the Siphon library so users can save entire web pages, clips from a page, highlighted sections, or specific links or images through drag and drop (Figure 7.3). Based on the type of content saved, Fuse will try and fetch additional metadata, such as the name and image for an article, or the authors and venue of a published PDF.

While Fuse is designed to allow users to save any size piece of information from the web into a lightweight structure, it still requires users to manually build and maintain

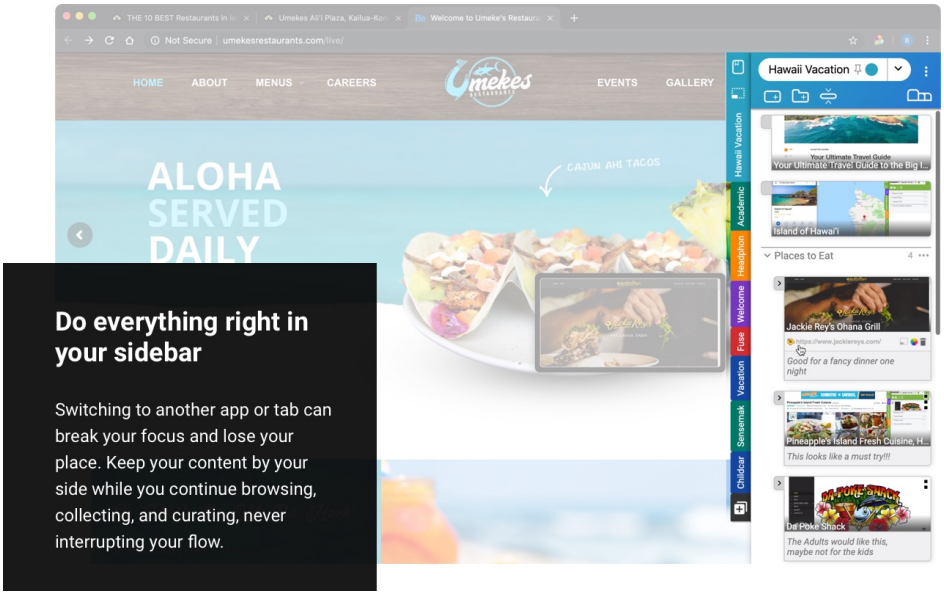


Figure 7.2: The Fuse Sidebar

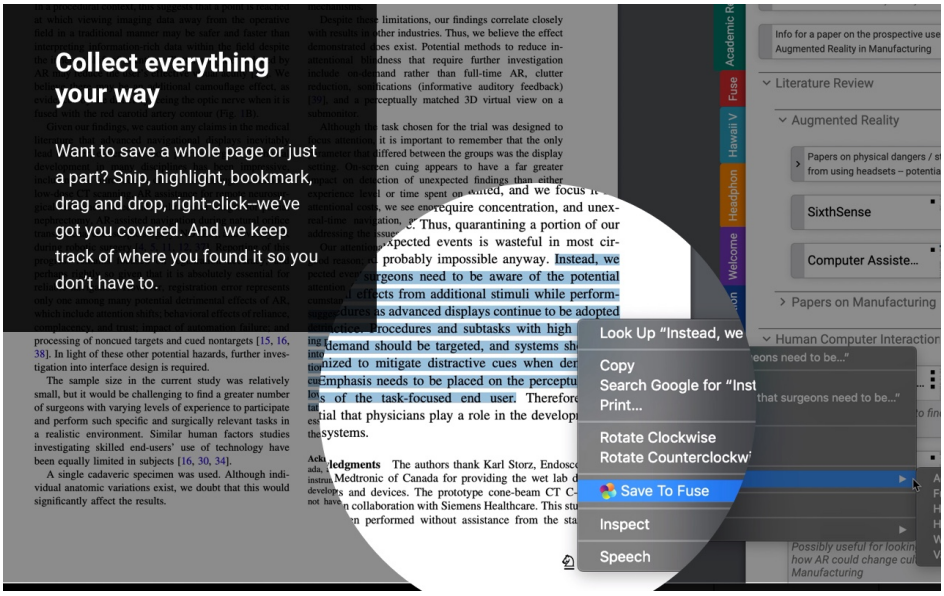


Figure 7.3: The Fuse system with Siphon Tools integration

that structure. This is where I see the benefit of the Distil smart categories in Fuse. One possible instantiation of this would be the introduction of smart folders into Fuse that would utilize the same mechanism presented in Distil to auto-categorize different items. Another alternative would be an additional structure, possibly a tagging system, where the system could automatically recommend valid tags for particular item, with a small confirmation on the user's part to assign those tags.

7.2 From Outline to Full Structure

In its current design, Fuse only supports a lightweight outline-based organization. While this is sufficient many smaller projects and is able to support hierarchical organizations well, for other sensemaking activities, it's insufficient. For example, product comparison – users will often make use of a tabular structure for performing comparison due to the cross referencing that needs to occur across the different options. Another example would be affinity diagramming. These diagrams are often utilized by individuals attempting to perform complex schema induction. These schemas are then utilized to infer problems, missing information, or limitations which can then be further explored or used for solution inspiration.

By developing these additional structures for users to mold the information from their Fuse trove, processes such as inference and comparison can be realized. Users can then take the output of those processes, and utilize them to drive the next stage of their sensemaking process. I have begun to develop a series of workspaces (Forge) for the Fuse system that allow users to take information out of their outline and restructure it into different formats. First is a free-form workspace, similar to the canvas provided by the IdeaMache system [98]. Users can freely position, resize, and draw on content from their outline, producing a collage of content and higher-level structure. Second is a document view, where users can construct documents with reference to the cards in their outline. Third is a table view, where users can use cards as either a row item (in the case of specific options, such as a dishwasher) or a cell item (like the clipped price of a dishwasher). Lastly, there is a kanban-style board, where users can make reorganizable lists of card, effectively generating alternative groupings from their outline.

While each of these workspaces exist in a basic form, the currently lack any way for users to quickly and efficiently transfer information into them without significant effort. One of the critical goals proposed by the dataflow-centric sensemaking model (Figure 7.1) is to decrease the friction required to transfer information from one artifact and phase to the next in support of the underlying mechanisms that need to occur. Therefore, I plan to explore different ways of marshaling data into these various view, such as some type of automatic content schema recognition, like that performed in Thresher [64].

7.3 Deployment

I plan to deploy Fuse to a large set of users, both as a way to validate the new features I am proposing, but additionally as a way to collect a useful dataset of in-depth sensemaking processes. Currently, most information regarding sensemaking behavior

exists around search log queries and browser behaviors [21, 49, 139, 144]. However, a significant set of activities occur outside of these contexts - such as note taking, final artifact generation, and collaboration. While Fuse in its current form is designed to streamline the sensemaking process, it also serves as a platform for collecting richer sensemaking behavior.

In order to collect this behavior, we plan to deploy Fuse to a large set of users, through both ad targeting, word-of-mouth, and online target user populations groups (academic researchers on Reddit, etc.). This is planned to be a staged deployment: our initial deployed version of the Fuse system will just incorporate the baseline Bento and Siphon features. Based on the success of that deployment, we will either tweak which features we include to either reduce the complexity of the system or to increase the value provided. Once we have a fairly large set of active users (500+), I plan to continue development of additional features, based on Distil and proposed Forge workspaces. These will also be deployed in stages, and using both usage metrics from the Fuse system, as well as a series of in-person lab studies, I will evaluate the effectiveness of those interventions.

7.4 Timeline of Completion

My goal is to complete my dissertation by May 2020. Below is a timeline of my plan:

- September 2019 - November 2019: Initial Deployment of the Fuse system with current Bento and Siphon integrations
- October 2019 - December 2019: Integration of Distil concepts into Fuse and initial small set test of Distil-based features
- November 2019 - February 2020: Development of the the enhanced Fuse workspaces
- February 2020 - April 2020: Deployment of new Fuse workspace feature
- March 2020 - May 2020: Write dissertation and defend

Bibliography

[1] Eytan Adar, Mira Dontcheva, James Fogarty, and Daniel S Weld. Zoetrope: interacting with the ephemeral web. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 239–248. ACM, 2008.

[2] Annette Adler, Anuj Gujar, Beverly L Harrison, Kenton O'hara, and Abigail Sellen. A diary study of work-related reading: design implications for digital reading devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 241–248. ACM Press/Addison-Wesley Publishing Co., 1998.

[3] Salman Ahmad, Alexis Battle, Zahan Malkani, and Sepander Kamvar. The jabberwocky programming environment for structured social computing. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 53–64. ACM, 2011.

[4] Erik M Altmann and J Gregory Trafton. Memory for goals: An activation-based model. *Cognitive science*, 26(1):39–83, 2002.

[5] Brian Amento, Loren Terveen, Will Hill, and Deborah Hix. Topicshop: enhanced support for evaluating and organizing collections of web sites. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pages 201–209. Citeseer, 2000.

[6] Corin R Anderson and Eric Horvitz. Web montage: A dynamic personalized start page. In *Proceedings of the 11th international conference on World Wide Web*, pages 704–712. ACM, 2002.

[7] J Anderson. Forrester Market Report: Consumer Behavior Online: A 2009 Deep Dive. http://www.forrester.com/go?docid=54327, 2009. Accessed: 2017-09-10.

[8] John R Anderson and Robert Milson. Human memory: An adaptive perspective. *Psychological Review*, 96(4):703, 1989.

[9] Salvatore Andolina, Khalil Klouche, Jaakko Peltonen, Mohammad Hoque, Tuukka Ruotsalo, Diogo Cabral, Arto Klami, Dorota Głowacka, Patrik Floréen, and Giulio Jacucci. Intentstreams: smart parallel search streams for branching exploratory search. In *Proceedings of the 20th international conference on intelligent user interfaces*, pages 300–305. ACM, 2015.

[10] Paul André, Aniket Kittur, and Steven P Dow. Crowd synthesis: Extracting categories and clusters from complex data. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 989–998. ACM, 2014.

[11] Paul André, Robert E Kraut, and Aniket Kittur. Effects of simultaneous and sequential work structures on distributed collaborative interdependent tasks. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 139–148. ACM, 2014.

[12] Rajiv Badi, Soonil Bae, J Michael Moore, Konstantinos Meintanis, Anna Zacchi, Haowei Hsieh, Frank Shipman, and Catherine C Marshall. Recognizing user interest and document value from reading and organizing activities in document triage. In *Proceedings of the 11th international conference on Intelligent user interfaces*, pages 218–225. ACM, 2006.

[13] Michelle Q Wang Baldonado and Terry Winograd. Sensemaker: an information-exploration interface supporting the contextual evolution of a user's interests. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 11–18. ACM, 1997.

[14] Nikola Banovic, Christina Brant, Jennifer Mankoff, and Anind Dey. Proactivetasks: the short of mobile device use sessions. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*, pages 243–252. ACM, 2014.

[15] Regina Barzilay, Kathleen R McKeown, and Michael Elhadad. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 550–557. Association for Computational Linguistics, 1999.

[16] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 313–322. ACM, 2010.

[17] Michael S Bernstein, Desney Tan, Greg Smith, Mary Czerwinski, and Eric Horvitz. Personalization via friendsourcing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 17(2):6, 2010.

[18] Michael S Bernstein, Jaime Teevan, Susan Dumais, Daniel Liebling, and Eric Horvitz. Direct answers for search queries in the long tail. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 237–246. ACM, 2012.

[19] Eric A Bier, Edward W Ishak, and Ed Chi. Entity quick click: rapid text copying based on automatic entity extraction. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*, pages 562–567. ACM, 2006.

[20] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, et al. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 333–342. ACM, 2010.

[21] Robert Capra. Hci browser: A tool for administration and data collection for studies of web search behaviors. In *International Conference of Design, User Experience, and Usability*, pages 259–268. Springer, 2011.

[22] Robert Capra, Gary Marchionini, Javier Velasco-Martin, and Katrina Muller. Tools-at-hand and learning in multi-session, collaborative search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 951–960. ACM, 2010.

[23] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM, 1998.

[24] Stuart K Card, George G Robertson, and William York. The webbook and the web forager: an information workspace for the world-wide web. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 111–ff. ACM, 1996.

[25] Pew Research Center. Generational differences in online activities. Report, July 2015. `http://www.pewinternet.org/2009/01/28/generational-differences-in-online-activities/`.

[26] Dana Chandler and Adam Kapelner. Breaking monotony with meaning: Motivation in crowdsourcing markets. *Journal of Economic Behavior & Organization*, 90:123–133, 2013.

[27] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[28] Joseph Chee Chang, Nathan Hahn, and Aniket Kittur. Supporting mobile sensemaking through intentionally uncertain highlighting. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 61–68. ACM, 2016.

[29] Joseph Chee Chang, Nathan Hahn, and Aniket Kittur. Supporting mobile sensemaking through intentionally uncertain highlighting. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 61–68, New York, NY, USA, 2016. ACM.

[30] Joseph Chee Chang, Nathan Hahn, Adam Perer, and Aniket Kittur. Searchlens: Composing and capturing complex user interests for exploratory search. In *Proceedings of the 2019 ACM 24th Annual Meeting of the Intelligent User Interfaces*, IUI'19, pages 498–509, 2019.

[31] Joseph Chee Chang, Aniket Kittur, and Nathan Hahn. Alloy: Clustering with crowds and computation. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3180–3191. ACM, 2016.

[32] Duen Horng Chau, Aniket Kittur, Jason I. Hong, and Christos Faloutsos. Apolo: Making sense of large network data by combining rich user interaction and machine learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 167–176, New York, NY, USA, 2011. ACM.

[33] Chen Chen, Simon T Perrault, Shengdong Zhao, and Wei Tsang Ooi. Bezelcopy: an efficient cross-application copy-paste technique for touchscreen smartphones. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, pages 185–192. ACM, 2014.

[34] Justin Cheng, Jaime Teevan, Shamsi T Iqbal, and Michael S Bernstein. Break it down: A comparison of macro-and microtasks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 4061–4064. ACM, 2015.

[35] Parmit K Chilana, Andrew J Ko, and Jacob O Wobbrock. Lemonaid: selection-based crowdsourced contextual help for web applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1549–1558. ACM, 2012.

[36] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1999–2008. ACM, 2013.

[37] Andy Cockburn, Saul Greenberg, Bruce McKenzie, Michael Jasonsmith, and Shaun Kaasten. Webview: A graphical aid for revisiting web pages. In *Proceedings of the OZCHI*, volume 99, pages 15–22, 1999.

[38] Shelia R Cotten and Sipi S Gupta. Characteristics of online and offline health information seekers and factors that discriminate between them. *Social science & medicine*, 59(9):1795–1806, 2004.

[39] Douglass R Cutting, David R Karger, Jan O Pedersen, and John W Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 318–329. ACM, 1992.

[40] Richard L Daft and Karl E Weick. Toward a model of organizations as interpretation systems. *Academy of management review*, 9(2):284–295, 1984.

[41] Hoa Trang Dang, Diane Kelly, and Jimmy J Lin. Overview of the trec 2007 question answering track. In *TREC*, volume 7, page 63, 2007.

[42] Brenda Dervin. *An overview of sense-making research: Concepts, methods, and results to date*. The Author, 1983.

[43] Brenda Dervin. From the mind's eye of the user: The sense-making qualitative-quantitative methodology. *Qualitative research in information management*, 9:61–84, 1992.

[44] Brenda Dervin. Sense-making theory and practice: an overview of user interests in knowledge seeking and use. *Journal of knowledge management*, 2(2):36–46, 1998.

[45] Jerry Dischler. Building for the next moment. `http://adwords.blogspot.com/2015/05/building-for-next-moment.html`, 2015.

[46] Mira Dontcheva, Steven M Drucker, Geraldine Wade, David Salesin, and Michael F Cohen. Collecting and organizing web content. In *Personal Information Management-Special Interest Group for Information Retrieval Workshop*, pages 44–47, 2006.

[47] Mira Dontcheva, Steven M Drucker, Geraldine Wade, David Salesin, and Michael F Cohen. Summarizing personal web browsing sessions. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 115–124. ACM, 2006.

[48] Laura Drăgan, Siegfried Handschuh, and Stefan Decker. The semantic desktop at work: Interlinking notes. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 17–24, New York, NY, USA, 2011. ACM.

[49] Patrick Dubroy and Ravin Balakrishnan. A study of tabbed browsing among mozilla firefox users. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 673–682. ACM, 2010.

[50] Günes Erkan and Dragomir R Radev. Lexrank: graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, pages 457–479, 2004.

[51] Leah Findlater, Alex Jansen, Kristen Shinohara, Morgan Dixon, Peter Kamb, Joshua Rakita, and Jacob O Wobbrock. Enhanced area cursors: reducing fine pointing demands for people with motor impairments. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 153–162. ACM, 2010.

[52] Kristie Fisher, Scott Counts, and Aniket Kittur. Distributed sensemaking: improving sensemaking by leveraging the efforts of previous users. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 247–256. ACM, 2012.

[53] Dennis A Gioia and Kumar Chittipeddi. Sensemaking and sensegiving in strategic change initiation. *Strategic management journal*, 12(6):433–448, 1991.

[54] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. Multi-document summarization by sentence extraction. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization-Volume 4*, pages 40–48. Association for Computational Linguistics, 2000.

[55] Vishal Gupta and Gurpreet Singh Lehal. A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence*, 2(3):258–268, 2010.

[56] Udo Hahn and Ulrich Reimer. Knowledge-based text summarization: Salience and generalization operators for knowledge base abstraction. *Advances in Automatic Text Summarization*, pages 215–232, 1999.

[57] Jaehyun Han and Geehyuk Lee. Push-push: A drag-like operation overlapped with a page transition operation on touch interfaces. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 313–322. ACM, 2015.

[58] Marti A. Hearst. Clustering versus faceted categories for information exploration. *Commun. ACM*, 49(4):59–61, April 2006.

[59] Marti A Hearst and Duane Degler. Sewing the seams of sensemaking: A practical interface for tagging and organizing saved search results. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval*, page 4. ACM, 2013.

[60] D Austin Henderson Jr and Stuart Card. Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Transactions on Graphics (TOG)*, 5(3):211–243, 1986.

[61] Ron R Hightower, Laura T Ring, Jonathan I Helfman, Benjamin B Bederson, and James D Hollan. Graphical multiscale web histories: a study of padprints. In *The Craft of Information Visualization*, pages 220–227. Elsevier, 2003.

[62] Ken Hinckley, Shengdong Zhao, Raman Sarin, Patrick Baudisch, Edward Cutrell, Michael Shilman, and Desney Tan. InkSeine. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI 07*. ACM Press, 2007.

[63] Johann Hofmann. xpath-dom. `https://github.com/johannhof/xpath-dom`, 2015–2019.

[64] Andrew Hogue and David Karger. Thresher: automating the unwrapping of semantic content from the world wide web. In *Proceedings of the 14th international conference on World Wide Web*, pages 86–95. ACM, 2005.

[65] Jeff Huang and Ryen W White. Parallel browsing behavior on the web. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, pages 13–18. ACM, 2010.

[66] David Huynh, Stefano Mazzocchi, and David Karger. Piggy bank: Experience the semantic web inside your web browser. In *International Semantic Web Conference*, pages 413–430. Springer, 2005.

[67] Shamsi T. Iqbal, Jaime Teevan, Dan Liebling, and Anne Loomis Thompson. Multitasking with play write, a mobile microproductivity tool. In *Proceedings of the 31st annual ACM symposium on User interface software and technology*. ACM, 2018.

[68] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[69] Jim Jones. Turkee ruby gem. `https://github.com/aantix/turkee`, 2013.

[70] David Jonker, William Wright, David Schroh, Pascale Proulx, Brian Cort, et al. Information triage with trist. In *2005 International Conference on Intelligence Analysis*, pages 2–4, 2005.

[71] Ece Kamar and Eric Horvitz. Light at the end of the tunnel: A monte carlo approach to computing value of information. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pages 571–578, Richland, SC, 2013. International Foundation for Autonomous Agents and Multiagent Systems.

[72] Eser Kandogan and Ben Shneiderman. Elastic windows: evaluation of multi-window operations. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 250–257. ACM, 1997.

[73] Victor Kaptelinin and Mary Czerwinski. *Beyond the desktop metaphor: designing integrated digital work environments*, volume 1. The MIT Press, 2007.

[74] Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J Guo, Robert C Miller, and Krzysztof Z Gajos. Crowdsourcing step-by-step information extraction to enhance existing how-to videos. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 4017–4026. ACM, 2014.

[75] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[76] Aniket Kittur, Susheel Khamkar, Paul André, and Robert Kraut. Crowdweaver: Visually managing complex crowd work. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, CSCW '12, pages 1033–1036, New York, NY, USA, 2012. ACM.

[77] Aniket Kittur and Robert E Kraut. Harnessing the wisdom of crowds in wikipedia: quality through coordination. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 37–46. ACM, 2008.

[78] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 1301–1318. ACM, 2013.

[79] Aniket Kittur, Andrew M Peters, Abdigani Diriye, and Michael Bove. Standing on the schemas of giants: socially augmented information foraging. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 999–1010. ACM, 2014.

[80] Aniket Kittur, Andrew M Peters, Abdigani Diriye, Trupti Telang, and Michael R Bove. Costs and benefits of structured information foraging. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2989–2998. ACM, 2013.

[81] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E Kraut. Crowdforge: Crowdsourcing complex work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 43–52. ACM, 2011.

[82] Aniket Kittur, Bongwon Suh, Bryan A Pendleton, and Ed H Chi. He says, she says: conflict and coordination in wikipedia. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 453–462. ACM, 2007.

[83] Gary Klein, Brian Moon, and Robert R Hoffman. Making sense of sensemaking 2: A macrocognitive model. *Intelligent Systems, IEEE*, 21(5):88–92, 2006.

[84] Gary Klein, Jennifer K Phillips, Erica L Rall, and Deborah A Peluso. A data–frame theory of sensemaking. In *Expertise out of context*, pages 118–160. Psychology Press, 2007.

[85] Khalil Klouche, Tuukka Ruotsalo, Luana Micallef, Salvatore Andolina, and Giulio Jacucci. Visual re-ranking for multi-aspect information retrieval. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, pages 57–66. ACM, 2017.

[86] Steffen Koch, Markus John, Michael Worner, Andreas Muller, and Thomas Ertl. VarifocalReader — in-depth visual analysis of large text documents. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1723–1732, dec 2014.

[87] Weize Kong and James Allan. Extending faceted search to the general web. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 839–848, New York, NY, USA, 2014. ACM.

[88] Jonathan Koren, Yi Zhang, and Xue Liu. Personalized interactive faceted search. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 477–486, New York, NY, USA, 2008. ACM.

[89] Bill Kules and Robert Capra. Creating exploratory tasks for a faceted search interface. In *Second Workshop on Human-Computer Interaction (HCIR 2008)*, 2008.

[90] Bill Kules, Robert Capra, Matthew Banta, and Tito Sierra. What do exploratory searchers look at in a faceted search interface? In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pages 313–322. ACM, 2009.

[91] Anand P Kulkarni, Matthew Can, and Bjoern Hartmann. Turkomatic: automatic recursive task and workflow design for mechanical turk. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 2053–2058. ACM, 2011.

[92] Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. Real-time captioning by groups of non-experts. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 23–34. ACM, 2012.

[93] Walter S Lasecki, Christopher D Miller, and Jeffrey P Bigham. Warping time for more effective real-time crowdsourcing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2033–2036. ACM, 2013.

[94] Walter S Lasecki, Christopher D Miller, Raja Kushalnagar, and Jeffrey P Bigham. Legion scribe: real-time captioning by the non-experts. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, page 22. ACM, 2013.

[95] Edith Law and Haoqi Zhang. Towards large-scale collaborative planning: Answering high-level search queries using human computation. In *AAAI*, 2011.

[96] Harlan Lebo. 2017 digital future project: Surveying the digital future. Market research report, Center for the Digital Future at USC Annenberg, 2017.

[97] Seungyon Claire Lee, Eamonn O'Brien-Strain, Jerry Liu, and Qian Lin. A survey on web use: How people access, consume, keep, and organize web content. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 619–628, New York, NY, USA, 2012. ACM.

[98] Rhema Linder, Nic Lupfer, Andruid Kerne, Andrew M. Webb, Cameron Hill, Yin Qu, Kade Keith, Matthew Carrasco, and Elizabeth Kellogg. Beyond slideware: How a free-form presentation medium stimulates free-form thinking in the classroom. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, C&C '15, pages 285–294, New York, NY, USA, 2015. ACM.

[99] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Turkit: human computation algorithms on mechanical turk. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 57–66. ACM, 2010.

[100] Kurt Luther, Casey Fiesler, and Amy Bruckman. Redistributing leadership in online creative collaboration. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 1007–1022, New York, NY, USA, 2013. ACM.

[101] Yoelle S Maarek, Michal Jacovi, Menachem Shtalhaim, Sigalit Ur, Dror Zernik, and Israel Z Ben-Shaul. Webcutter: a system for dynamic and tailorable site mapping. *Computer networks and ISDN systems*, 29(8-13):1269–1279, 1997.

[102] Richard Mander, Gitta Salomon, and Yin Yin Wong. A "pile" metaphor for supporting casual organization of information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 627–634. ACM, 1992.

[103] Inderjeet Mani and Eric Bloedorn. Multi-document summarization by graph search and matching. *arXiv preprint cmp-lg/9712004*, 1997.

[104] Gary Marchionini. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.

[105] Catherine C. Marshall and Sara Bly. Saving and using encountered information: Implications for electronic periodicals. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 111–120, New York, NY, USA, 2005. ACM.

[106] Catherine C Marshall and Frank M Shipman III. Spatial hypertext and the practice of information triage. In *Proceedings of the eighth ACM conference on Hypertext*, pages 124–133. ACM, 1997.

[107] Kathleen McKeown, Judith Klavans, Vasileios Hatzivassiloglou, Regina Barzilay, and Eleazar Eskin. Towards multidocument summarization by reformulation: Progress and prospects. In *AAAI/IAAI*, pages 453–460, 1999.

[108] James McKinney. Tfidfsimilarity ruby gem. `https://github.com/jpmckinney/tf-idf-similarity`, 2015.

[109] Frances J Milliken. Perceiving and interpreting environmental change: An examination of college administrators' interpretation of changing demographics. *Academy of management Journal*, 33(1):42–63, 1990.

[110] Neema Moraveji, Daniel Russell, Jacob Bien, and David Mease. Measuring improvement in user search performance resulting from optimal search tips. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 355–364. ACM, 2011.

[111] Dan Morris, Meredith Ringel Morris, and Gina Venolia. Searchbar: a search-centric web history for task resumption and information re-finding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1207–1216. ACM, 2008.

[112] Meredith Ringel Morris, AJ Bernheim Brush, and Brian R Meyers. Reading revisited: Evaluating the usability of digital display surfaces for active reading tasks. In *Horizontal Interactive Human-Computer Systems, 2007. TABLETOP'07. Second Annual IEEE International Workshop on*, pages 79–86. IEEE, 2007.

[113] Meredith Ringel Morris and Eric Horvitz. Searchtogether: an interface for collaborative web search. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 3–12. ACM, 2007.

[114] Meredith Ringel Morris, Jarrod Lombardo, and Daniel Wigdor. Wesearch: supporting collaborative search and sensemaking on a tabletop display. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 401–410. ACM, 2010.

[115] Michael Nebeling, Alexandra To, Anhong Guo, Adrian A de Freitas, Jaime Teevan, Steven P Dow, and Jeffrey P Bigham. Wearwrite: Crowd-assisted writing from smartwatches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3834–3846. ACM, 2016.

[116] Gergana Y Nenkov, Maureen Morrin, Andrew Ward, Barry Schwartz, John Hulland, et al. A short form of the maximization scale: Factor structure, reliability and validity studies. *Judgment and Decision Making*, 3:371–388, 2008.

[117] Antti Oulasvirta and Pertti Saariluoma. Surviving task interruptions: Investigating the implications of long-term working memory theory. *International Journal of Human-Computer Studies*, 64(10):941–961, 2006.

[118] Aditya Parameswaran, Ming Han Teh, Hector Garcia-Molina, and Jennifer Widom. Datasift: An expressive and accurate crowd-powered search toolkit. In *First AAAI Conference on Human Computation and Crowdsourcing*, 2013.

[119] Sharoda A Paul and Meredith Ringel Morris. Cosense: enhancing sensemaking for collaborative web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1771–1780. ACM, 2009.

[120] Sharoda A Paul and Madhu C Reddy. Understanding together: sensemaking in collaborative information seeking. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 321–330. ACM, 2010.

[121] Jaakko Peltonen, Kseniia Belorustceva, and Tuukka Ruotsalo. Topic-relevance map: Visualization for improving search result comprehension. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, pages 611–622. ACM, 2017.

[122] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[123] Peter Pirolli and Stuart Card. Information foraging. *Psychological review*, 106(4):643, 1999.

[124] Peter Pirolli and Stuart Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, volume 5, pages 2–4, 2005.

[125] Brian T Ratchford, Myung-Soo Lee, and Debabrata Talukdar. The impact of the internet on information search for automobiles. *Journal of Marketing research*, 40(2):193–209, 2003.

[126] Daniela Retelny, Sébastien Robaszkiewicz, Alexandra To, Walter S Lasecki, Jay Patel, Negar Rahmati, Tulsee Doshi, Melissa Valentine, and Michael S Bernstein. Expert crowdsourcing with flash teams. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 75–85. ACM, 2014.

[127] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[128] Jakob Rogstadius, Vassilis Kostakos, Aniket Kittur, Boris Smus, Jim Laredo, and Maja Vukovic. An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. In *ICWSM*, 2011.

[129] Volker Roth and Thea Turner. Bezel swipe: conflict-free scrolling and multiple selection on mobile touch screen devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1523–1526. ACM, 2009.

[130] Daniel M. Russell, Mark J. Stefik, Peter Pirolli, and Stuart K. Card. The cost structure of sensemaking. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, pages 269–276, New York, NY, USA, 1993. ACM.

[131] Frank M Shipman III, Haowei Hsieh, Preetam Maloor, and J Michael Moore. The visual knowledge builder: a second generation spatial hypertext. In *Proceedings of the 12th ACM conference on Hypertext and Hypermedia*, pages 113–122. ACM, 2001.

[132] Jeffrey Stylos and Brad A Myers. Mica: A web-search tool for finding api components and examples. In *Visual Languages and Human-Centric Computing (VL/HCC'06)*, pages 195–202. IEEE, 2006.

[133] Jeffrey Stylos, Brad A Myers, and Andrew Faulring. Citrine: providing intelligent copy-and-paste. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 185–188. ACM, 2004.

[134] Atsushi Sugiura and Yoshiyuki Koseki. Internet scrapbook: Automating web browsing tasks by demonstration. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*, UIST '98, pages 9–18, New York, NY, USA, 1998. ACM.

[135] Craig S Tashman and W Keith Edwards. Active reading and its discontents: the situations, problems and ideas of readers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2927–2936. ACM, 2011.

[136] Craig S Tashman and W Keith Edwards. Liquidtext: a flexible, multitouch environment to support active reading. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3285–3294. ACM, 2011.

[137] Linda Tauscher and Saul Greenberg. Revisitation patterns in world wide web navigation. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 399–406. ACM, 1997.

[138] Jaime Teevan, Susan T Dumais, and Zachary Gutt. Challenges for supporting faceted search in large, heterogeneous corpora like the web. *Proceedings of HCIR*, 2008:87, 2008.

[139] Jaime Teevan, Amy Karlson, Shahriyar Amini, A. J. Bernheim Brush, and John Krumm. Understanding the importance of location, time, and people in mobile local search behavior. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, pages 77–80, New York, NY, USA, 2011. ACM.

[140] Jaime Teevan, Daniel J Liebling, and Walter S Lasecki. Selfsourcing personal tasks. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, pages 2527–2532. ACM, 2014.

[141] Etherpad lite. `https://github.com/ether/etherpad-lite`, 2015.

[142] Notion - the all-in-one workspace. `https://www.notion.so/`, 2016. Accessed: 2018-04-10.

[143] Daniel Tunkelang. Faceted search. *Synthesis lectures on information concepts, retrieval, and services*, 1(1):1–80, 2009.

[144] Sarah K Tyler and Jaime Teevan. Large scale query log analysis of re-finding. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 191–200. ACM, 2010.

[145] Rajan Vaish, Keith Wyngarden, Jingshu Chen, Brandon Cheung, and Michael S Bernstein. Twitch crowdsourcing: crowd contributions in short bursts of time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3645–3654. ACM, 2014.

[146] Max G Van Kleek, Michael Bernstein, Katrina Panovich, Gregory G Vargas, David R Karger, and MC Schraefel. Note to self: examining personal information keeping in a lightweight note-taking tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1477–1480. ACM, 2009.

[147] Stephen Voida, Elizabeth D Mynatt, and W Keith Edwards. Re-framing the desktop interface around the activities of knowledge work. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 211–220. ACM, 2008.

[148] Qing Wang and Huiyou Chang. Multitasking bar: prototype and evaluation of introducing the task concept into a browser. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 103–112. ACM, 2010.

[149] Karl E Weick. Reduction of cognitive dissonance through task enhancement and effort expenditure. *The Journal of Abnormal and Social Psychology*, 68(5):533, 1964.

[150] Karl E. Weick. *Sensemaking in organizations*, volume 3. Sage, 1995.

[151] Ryen W White, Mikhail Bilenko, and Silviu Cucerzan. Studying the use of popular destinations to enhance web search interaction. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 159–166. ACM, 2007.

[152] Ryen W White, Bill Kules, Steven M Drucker, et al. Supporting exploratory search, introduction, special issue, communications of the acm. *Communications of the ACM*, 49(4):36–39, 2006.

[153] Ryen W White and Resa A Roth. Exploratory search: Beyond the query-response paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 1(1):1–98, 2009.

[154] William Wright, David Schroh, Pascale Proulx, Alex Skaburskis, and Brian Cort. The sandbox for analysis: concepts and methods. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 801–810. ACM, 2006.

[155] Dongwook Yoon, Nicholas Chen, and François Guimbretière. Texttearing: opening white space for digital ink annotation. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pages 107–112. ACM, 2013.

[156] Oren Zamir and Oren Etzioni. Grouper: a dynamic clustering interface to web search results. *Computer Networks*, 31(11-16):1361–1374, 1999.

[157] Amy X. Zhang, Michele Igo, Marc Facciotti, and David Karger. Using student annotated hashtags and emojis to collect nuanced affective states. In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale*, L@S '17, pages 319–322, New York, NY, USA, 2017. ACM.

[158] Haoqi Zhang, Edith Law, Rob Miller, Krzysztof Gajos, David Parkes, and Eric Horvitz. Human computation tasks with global constraints. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 217–226. ACM, 2012.

[159] Pengyi Zhang and Dagobert Soergel. Towards a comprehensive model of the cognitive process and mechanisms of individual sensemaking. *Journal of the Association for Information Science and Technology*, 65(9):1733–1756, 2014.

[160] Yuxiang Zhu, David Modjeska, Daniel Wigdor, Shengdong Zhao, et al. Hunter gatherer: interaction support for the creation and management of within-web-page collections. In *Proceedings of the 11th international conference on World Wide Web*, pages 172–181. ACM, 2002.