Interactivity, Control, and Visualization in Decision Making on the Web

Patrick Niebrzydowski

MASTERARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Interactive Media

in Hagenberg

im September 2017

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Hagenberg, September 25, 2017

Patrick Niebrzydowski

Contents

Abstract

Decisions on a fundamental level come down to retrieving relevant information based on identified criteria to help in coming to a final conclusion. With the multitude of information available at our fingertips, we constantly turn to the Web to help make decisions. Unfortunately, finding the relevant information is an increasingly difficult proposition due to the growing number of resources. As a result, numerous solutions have been presented to mitigate this information overload. Whether looking for a flight, choosing a university to attend, or finding a restaurant for dinner, users now have the ability to search libraries in seconds and receive personalized recommendations based on complex algorithms.

However, as these systems are developed, it is important to ensure that the user is included in the process. Many systems use long-term learning techniques to improve their algorithms, providing relevant results based on past behaviors. However, the experience a user receives during specific decision processes should also be considered. The criteria used to make a decision varies greatly from one user to another, and from one situation to another. Being able to properly communicate these criteria to the system is an important aspect of receiving useful results. Additionally, presenting the results in a way that is easy to understand can improve the process immensely.

Specifically aimed at individual decisions, a prototype was developed to investigate how user control and information visualization affect decision making on the Web. The prototype was designed to encourage user interaction and leverage visualization techniques to improve the decision process. The tool was then evaluated through a user study and survey. This study indicated that the general attitude was positive toward both criteria control and visual results.

Kurzfassung

Im Entscheidungsprozess kommt es grundsätzlich darauf an, relevante Informationen basiert auf ausschlaggebenden Kriterien zu sammeln, um einen endgültigen Entschluss zu fassen. Ständig von Informationen umgeben, wenden wir uns tagtäglich an das Internet, um uns dabei zu helfen Entscheidungen zu fällen. Durch die wachsende Menge an Resourcen wird der Prozess des Filterns relevanter Inhalte zusehends schwieriger. Zahlreiche Lösungen wurden daher bereits präsentiert die darauf abzielen diese Informationsflut überschaubarer zu gestalten. Ob auf der Suche nach einem Flug, der Auswahl einer Universität, oder eines Restaurants zum Abendessen, Benutzer/innen haben heutzutage die Möglichkeit Datenbanken in Sekundenschnelle zu durchsuchen und mit Hilfe komplexer Algorithmen personalisierte Vorschläge zu erhalten.

Obwohl diese Systeme zahlreich entwickelt werden, ist es wichtig sicherzustellen dass der/die Endnutzer/in in den Entscheidungsprozess integriert wird. Viele Lösungen setzen langfristige, maschinelle Lernverfahren ein um personalisierte Ergebnisse bieten zu können, was jedoch oftmals vergessen wird, ist die Erfahrungen des/der Benutzers/-Benutzerin in den Prozess miteinzubeziehen. Entscheidungskriterien unterscheiden sich stark von Anwender/in zu Anwender/in und sind situationsspezifisch. Die Möglichkeit diese Kriterien klar zu kommunizieren und durch den/die Benutzer/in beinflussen zu lassen ist daher ein wichtiger Aspekt, um relevante Ergebnisse zu bieten. Zusätzlich verspricht die eingängliche und verständliche Präsentation der Resultate die Verbesserung des Prozesses.

Mit besonderer Ausrichtung auf individuelle Entscheidungen wurde ein Prototyp entwickelt, der den Einfluss von Benutzerkontrolle und Datenvisualisierung im Bezug auf online-basierte Entscheidungsprozesse untersucht. Die resultierende Applikation zielt darauf ab Benutzerinteraktion zu fördern und Visualisierungtechniken wirksam einzusetzen, um den Entscheidungsprozess zu vereinfachen. Der Prototyp wurde in der Folge durch eine Benutzerstudie evaluiert. Die Ergebnisse dieser Untersuchung zeigen eine positive Haltung gegenüber der Kriterienkontrolle und der gewählten Visualisierung der Resultate.

Chapter 1

Introduction

With the development of new technologies, the amount of information available on the Web is continuing to increase exponentially. Consequently, people are often overcome and overwhelmed, making it difficult to identify the information they are seeking. When it comes to decisions, being able to sort through all the available knowledge and identifying which data is most relevant and useful can be a challenge. To address this problem, numerous applications seek to help users in mitigating this and provide them with the tools needed to come to the best possible conclusion.

Two primary types of tools used on the Web to aide users in making decisions are Search Systems and Recommender Systems. Search Systems allow input of specific queries and return relevant results based on the terms. Additionally, some Search Systems offer specific predefined or recommended attributes called facets, or the ability to filter results based on certain qualifications. Recommender Systems aim instead to provide relevant information based on other factors. These may include things such as other similar users or products, profile information, or explicit user input. Ultimately, both of these systems seek to make the retrieval of relevant information simpler and help users through the decision process.

As time goes on, new features are constantly added to help discover relevant information. Additionally, the way that data is actually presented to the user can have a significant effect on user experience and ultimately, user satisfaction. As these tools become more intelligent at identifying and filtering out information according to various attributes and characteristics, we have to ask whether the systems are appropriately taking the user into account. How do developers maintain a balance between implicit suggestions and explicit control for the users? How can it be ensured that users are not only receiving the most accurate or effective results, but also the ones that actually match what they were looking for in the first place?

1.1 Motivation

As tools have become more powerful at collecting information about users and their preferences, systems have also become more automated and robust in terms of providing information. While this is a useful functionality, it increases the potential that these systems become too focused on providing accurate or objectively effective data retrieval and recommendations. However, it is important when creating a decision system to consider individual differences between various users, as well as changes in criteria for different scenarios. At the core of any decision process should be the decision maker. For this reason, it is vital that in systems aimed to aide users in making decisions, a focus on user experience and satisfaction is emphasized. Rather than trying to provide what may be considered “good” information, the goal should be to provide “useful” and “relevant” knowledge which matches what the user is seeking. The motivation behind this study is to investigate how decision support systems can avoid pitfalls that lead to ignoring the needs of the user due to increasing automation during the process, and rather empower the user to feel a larger sense of control over their decisions.

1.2 Goals

The primary goal of this paper is to identify new and interesting ways to provide relevant results during the information retrieval process. Rather than focus on the retrieval aspect itself, or on the artificial intelligence and machine learning techniques that many systems seek to perfect, the study is aimed specifically at how the experience could be improved for the user. It is also aimed at improving the individual sessions, rather than the overall long-term abilities of an application. For this reason, the primary targets of the investigation are in the generation and communication of decision criteria, as well as the way that information is presented to the user.

Due to the amount of information overload that users are forced to endure, being able to narrow down that data to the most relevant pieces is central in improving the user experience during the decision process. Additionally, systems can improve the process by presenting it in a simple and easy-to-understand way. Based on much of the previous research in the domain of decision making on the Web, a gap was identified in two primary areas: control of decision criteria, and visualization of presented information.

First, the ability to control and communicate criteria to a system is generally limited. Search Systems are usually quite open-ended, and also typically offer the ability to only view results based on single criteria or complex search strings. Other applications tend to require exploration through various defined criteria. These can be useful in narrowing down a larger data set, but also limit the users to those criteria presented. Additionally, there are not many systems which allow decision attributes to be factored into the results in different ways. However, when people make decisions, they typically consider multiple criteria and those criteria may not be equally important to the user. One goal of this research is to see how users can influence their decision criteria, especially within a single decision process, to better allow the system to find relevant results.

Second, though decision systems generally provide a way to narrow down the vast amount of data available, there is still often a large amount of information presented. Searches typically provide long lists of results, often containing an overload of text. Recommendations are often oversimplified and confused users as to why options are being suggested. Additionally, users often do not have much influence or immediate interactivity with these systems. One question this paper seeks to answer is whether using visual techniques can help improve experiences in parsing information before making a decision. As visualizations are commonly used to help simplify large sets of information, applying visual result displays into the decision process is a logical step.

1.3 Structure

As an introduction to the core concepts explored, Chapter 2 seeks to define and identify key foundations and terms used throughout the thesis. Chapter 2 is split into two main portions. The first section discusses general concepts around decision making and user experiences during the decision process, whereas the second focuses more specifically on Web tools that support this process. The discussion of these tools focus primarily on two major types of decision support systems: Search Systems and Recommender Systems.

Chapter 3 contains relevant existing work which helps support these concepts. The past research outlined in Chapter 3 provides the base foundation for many of the decisions made throughout this paper. It explores numerous past surveys of decision systems as well as other original systems which are directly related to the core concepts discussed during this thesis.

Outlined in Chapter 4 is the general design of the prototype. This contains information about the motivations and goals of the study, as well as the structure of the interface itself. Additionally discussed is the planning of the development process and future extension possibilities. Once the general structure is outlined, a more detailed discussion into the technical implementation of the system is presented in Chapter 5. More specifically, it describes information about the code itself, system and component architecture, libraries used, as well as the build and deployment process.

An evaluation of the created prototype is presented in Chapter 6. Here, objectives and goals of the evaluation are introduced, as well as the overall design and execution details of the study. Additionally, results from the study are presented and discussed, with specific attention to detail regarding the core concepts of interactivity, control, and visualization.

Finally, Chapter 7 presents a final overview of the overall research, and a more general interpretation of the results. Additionally discussed is how those results relate to the past research and how they can be applied to potential future explorations.

Chapter 2

Foundations and Term Definition

This chapter aims to introduce key topics in the area of decision making, specifically as it relates to Web-based systems. The first portion will cover more general topics related to the psychology of decisions themselves and how they relate to technology and the Web. The second portion will outline topics more specifically related to tools available on the Web.

2.1 Decision Making

The concept of decision making covers the entire psychological process that people undergo when making a decision. Theories in the field generally investigate the effect that outside information has on the final decision quality [3]. This thesis focuses primarily on the intricacies of information gathering to support the final resolution.

2.1.1 Psychological Background

Though this thesis focuses on how tools on the Web can help users with choices, it is important to first understand some aspects of the process itself. Knowledge of how humans approach things from a psychological standpoint provides a broader insight into the design of technology to aid users in such a task.

Information Gathering: For nearly every choice one makes, some element of input enters into the equation. Though many evaluations can be made using preexisting information, others require external research to complete the process. As a result, users often turn to tools on the Web to help facilitate gathering relevant data.

Information Overload: In some cases, decisions can be hindered through excessive collection of information. This is typically referred to as information overload. The problem has become especially prevalent in recent years, due to the plethora of on-demand material available via the Web.

2.1.2 User Experience

Choices are often stressful enough on their own, so tools that aim to aid the decision-making process must pay special attention to usability and other comfort factors. From a design perspective, it is important to provide end users with a positive experience while using an application. This thesis looks especially at three particular areas which these types of tools often focus their attention on: interactivity, visualization, and control.

Interactivity: Through the maturation of the Web, tools are constantly employing communication techniques that focus on two-way conversation between user and machine. Instead of simply ingesting static information, users are increasingly encouraged to immerse themselves in the system. In this sense, interactivity is defined by the amount of involvement the user has during the process of information retrieval. In highly interactive systems, the content available is more dynamic, and the user is encouraged to explore rather than absorb.

Control: Though control and interactivity overlap in many ways, a distinct difference separates them. While interactivity focuses on the response of the system, control is about the amount of influence a user has over the content itself. In the context of decision-making systems, this means that the user can not only seek out existing information, but form new results through their own actions. For example, this could be as basic as choosing to filter certain result sets, or through more complex adjustment of calculation algorithms. How much control a user has over information can affect the confidence level before making a choice, as well as the satisfaction level after.

Visualization: Information can be presented in a multitude of different methods. One common way to display data, especially complex sets, is through visual techniques such as graphs, charts, and images. From a user perspective, providing visual support can often simplify information making it easier to understand, but it can also have the opposite effect. This thesis aims to identify how visual techniques, used in combination with interaction and control can change the way users view information on the web.

2.2 Decision Support Tools

An incredible breadth of knowledge is accessible through just a few clicks via the Web. However, the danger of overload is always looming, and sifting through the deluge of information can be intimidating. To help mitigate this, numerous solutions exist which help users detect that which is most important and relevant to their specific exploration. This paper focuses on tools which can be divided into two primary categories: Search Systems and Recommender Systems.

2.2.1 Search Systems

Likely the most basic type of decision aide on the Web is search. While the most basic example of this is a simple text query which returns a result set, it also encompasses a wide array of systems which help users to wade through the vast amount of information available. Though search in general applies specifically to the information gathering process, many decision support systems implement search strategies. Three particular methods within search that are often used in this field are search aggregation, Faceted Search, and filtering.

Search Aggregation: When operating in a specific domain, search aggregation is a way to increase the value of data provided. Search results may be combined from multiple sources to create a more robust data set. Aggregation can also refer to joining multiple queries to retrieve data from a single source. These methods can be beneficial, as a single entry point can retrieve data from multiple end points, saving the user time and effort from having to visit the different searches individually.

Faceted Search: One of the earlier methods for searching the Web was through Faceted Search (or Faceted Navigation/Browsing). This method allows users to dig deeper into a result set based on defined, concrete attributes. This allows the user to discover relevant results that apply to the specific domain in which the search is taking place. As each term is explored, the context of the search is redefined within those results.

Filtering: A feature that often is implemented to provide additional user control of a search is filtering. This allows users to choose a specific attribute or property which results much match in order to be displayed. By adding one or more filters to a search query, the result set can be parsed to remove information that is not relevant to the selected options.

2.2.2 Recommender Systems

Numerous systems exist which seek to provide solutions that rather than being an objective best option, instead aim for relevancy to the user. These Recommender Systems are especially common in consumer markets and the e-Commerce domain. Rather than the user seeking out information which is relevant, the systems utilize data such as profiles, trends, or product similarity to provide a recommendation which best fits the users' needs.

2.2.3 Types of Recommender Systems

Numerous types of Recommender Systems exist, each making use of different data and using different kinds of algorithms to generate recommendations. Historically, these are divided into three primary categories: Collaborative Filtering (CF), Content-Based (CB), and Knowledge-Based (KB). However, each type exhibits certain strengths and weaknesses, which has lead to an increased amount of hybrid approaches, such as social, fuzzy, context, and group systems [23].

Collaborative Filtering (CF): Recommendations generated using CF are primarily based on similarity in taste. One user receives items that other users with similar interests have also liked.

Content-Based (CB): Recommendations generated using CB systems are primarily based on similarity of products. These provide users with items that are similar to others which the same user has liked in the past.

Knowledge-Based (KB): Recommendations generated using KB systems are primarily based on matching between the product and user. Instead of focusing on similarity between users or items, these systems seek to find items which possess features that best match the user's needs and wants.

2.2.4 Explanation

One major aspect of Recommender Systems that has grown immensely over the past few years is explanation. This is when the system attempts to generate useful feedback to the user as to why items have been suggested. This provides additional information which allows the user to better assess whether the items actually fulfill their desires.

Goals

When a system provides explanation of recommendations, it seeks to improve the overall effectiveness of the results. Of course, as decision making is a highly subjective process, judging the effectiveness of a result is not necessarily consistent between users or systems. Typically, Recommender Systems using explanation seek to meet at least one of seven goals, shown in Table 2.1.

Efficiency	Reducing the time used to complete a task
Effectiveness	Helping the users make better decisions
Persuasiveness	Changing the user's buying behavior
Transparency	Explaining why a particular recommendation is made
Satisfaction	Increasing usability and enjoyment
Scrutability	Making the system's model more correctable
Trust	Increasing the user's confidence in the system

Table 2.1: Goals for explanations in Recommender Systems, taken from [8].

2.2.5 Ranking

A common tool on the Web to help users make choices is providing ranked lists. Rankings allow a user to view a specific set of data in an order that describes which item best matches the criteria. This order can be defined in many ways. For example, search results could be ranked by relevance using matching algorithms. Recommender Systems often provide lists that show which items best fit a user's implicit or explicit criteria. Additionally, these lists could simply be based on raw data, such as expert opinion, public polls, or statistical facts.

Chapter 3

State of Art

3.1 Decision Making Tools

As the Internet is growing, an increasing amount of information has become available. Unfortunately, this also gives rise to an increase in information overload and makes it more difficult for users to find the relevant information that they seek. When processing information, people try to apply different strategies, but they often process even identical objective stimuli differently [3]. To combat this, numerous systems and techniques are used to help with information retrieval and support the decision-making process.

Unfortunately, users tend to be bad at finding things on the web, a trend which has gotten worse over time [37]. As a result, providing a system which helps users during the process is an important challenge. This paper aims to investigate how decision-making tools on the Web can operate on a more personalized and immediate level.

3.1.1 Search

The simplest and most widespread tools used for information retrieval and decision making on the Web are in the domain of search. In searches, users can provide a query which will return the most relevant set of results. Of course, one challenge of this process is to identify which data is the most interesting for the user. Search queries can often present thousands of pages, but only a much smaller subset is truly relevant [16]. In addition to a rapidly growing amount of information to parse, some common issues in finding relevant information from search engines include: queries that are often short and ambiguous, search results that are poorly organized, and differences between user goals and expectations [10]. Though numerous studies show that most users turn to search, they do not really know how to use it [37]. Finding ways to optimize this process and provide more personalized data would improve the user experience.

In many domains, users seek to retrieve the most important responses. These are referred to as *top-k queries.* A common way of identifying the most relevant results in a top-k query is using a scoring function. Typically, these functions include multiple factors which comprise the overall score [14]. This allows the system to take into account a variety of influences to decide which information to prioritize. Unfortunately, e-commerce systems using ranked queries commonly order results according to a single attribute. Therefore, users are responsible for manually examining the results across multiple properties [12].

3.1.2 Recommendations

One of the most popular trends in Web tools for decision making is through personalized recommendations. The core concept surrounding Recommender Systems is to match a user to the most suitable items based on the user and item information as well as their interaction [23]. Typically, most of the recommendation algorithms operate in the background and provide little direct information or control about how the system works. As a result, recent research has investigated how to help provide transparency and expose the reasoning behind a recommendation [32].

Many different approaches are used to generate these recommendations. The three primary methods are Content-based (CB), Collaborative Filtering (CF), and Knowledge-based (KB). However, due to the varying strengths and weakness, numerous other hybrid approaches are consistently appearing. CB systems generally focus on long-term and large-data methods such as statistical learning, machine learning, and historical user models. CF techniques are instead primarily based on similarities, either between users or items [23]. However, this creates a weakness in that changing desires from one session to the next are often ignored.

3.1.3 Aggregation

A primary concern with information overload on the Web is the plethora of sources from which it comes. Even when a user has a well-defined idea of what to look for, it can be difficult and time-consuming to find the information. To improve efficiency of this process, many systems utilize ways to aggregate information from multiple sources.

How to perform aggregation of data from various sources is a problem that is well-ingrained within the field of information retrieval. It can be especially difficult to handle information coming from different search media. One potential solution to this is to look at aggregated search as a way to retrieve a unified representation of data from a single query across varying search spaces and sources [24]. Though the data structures may vary across result sets, relationships can often be inferred or defined to ease this process. A related area is the technique of schema matching. For proper aggregation of result sets when the data structure varies, a matching component must be implemented. One fundamental piece of this is through schema matching, which takes two schemas as input and outputs a mapping between elements which correspond to each other [27]. Though schema matching is generally associated with database applications, it can also be an important factor in accessing multiple Application Programming Interfaces (APIs).

When searching for products on the Web, users are often forced to switch between numerous searches to find those products which fit all their criteria. While performing a keyword search, users are often required to navigate through result pages which may not be entirely relevant. By combining multiple related queries, more relevant results could be retrieved [20]. Additionally, aggregating same-keyword queries from multiple sets can provide more relevant results. Though some domain-specific engines exist, they often are limited in the properties which can be searched. Some well-known examples of these include Shopping.com[1], Google Products[2], and Shopzilla[3] [33]. Aggregations like this often utilize APIs from other shops, but these are not always available for a proper aggregation. One proposed approach to battle this gap is through semantic search, in which the aggregation system scrapes information from various sources.

3.2 Decision Factors and User Satisfaction

The psychological process of making a decision differs from person. For example, a distinction can be made between maximizers and satisficers. Maximizers need to know they made the best decision that could have been made. Satisficers, however, aim to find something good enough to meet their criteria and not worry about possible better options [30]. Schwartz theorizes that satisficers are using a more efficient strategy and should be more likely to be satisfied with their decisions. However, Knijnenburg et al found that maximizers were actually more likely to maintain choice satisfaction than satisficers [18]. Of course, truly knowing whether the absolute best decision was made is nearly impossible in most cases. This is especially true when considering the wealth of knowledge available on the Web.

As a result, providing relevant information or products to individual users becomes important. Identifying user preferences and constraints is a challenge that is daunting for any decision support tool. Factors can be detected implicitly via statistical and machine-learning methods, or collected explicitly through user profiles and search mechanisms. However, it is almost always the case that consumers make decisions based on more than one factor. Ultimately, consumers will make decisions based on their own criteria, and both objective and subjective factors can impact the weighting of these criteria [21].

Unfortunately, specifying how various attributes are weighted can be very difficult, as consumers are not particularly well trained in existing methods. One proposed solution to this problem is using *Multi-Attribute Value Theory*-based systems [29]. However, identifying and weighting the attributes properly often requires a large amount of cognitive effort, making it difficult to apply in contexts like e-commerce in which users are unable or unwilling to spend a lot of time on their decision. Additionally, it is possible that a user may not know their goal in advance of making the decision [3].

3.2.1 Decision Factors and User Satisfaction in Search

While aggregation is a common technique in search systems, accommodating different criteria can be difficult. Often, the only option for assessing multiple criteria in search is through multiple searches, or through complex aggregation of terms within a single search field.

Query generation is one of the major challenges when it comes to searches. Systems generally extract and compute query weighting for multiple query terms. However, in some systems, the user is also able to control this as well. Unfortunately, this adds another layer of complexity, as users have been shown to not be particularly good at determining importance of query terms [1].

[1]http://uk.shopping.com
[2]Now called Google Shopping: https://www.google.com/shopping
[3]http://http://www.shopzilla.com/

3.2.2 Decision Factors and User Satisfaction in Recommendations

Historically, Recommender Systems are deemed as effective based on a few factors. Most systems within the e-commerce domain focus primarily either on probability of the user purchasing the suggested item, or on sensitivity of purchase probability as a result of the recommendation [15]. From a business standpoint, this appears rather obvious on the surface.

However, evaluation of what makes a successful or effective recommendation is not necessarily universal. Since recommendations are meant to be personalized by definition, they must not only match the user's needs, but should only recommend items which maintain a positive sentiment even after the decision has been made [15].

To achieve a high level of satisfaction, it is important to take into concern each user's individual preferences and constraints. To achieve this, it is important to identify the key attributes or values which each consumer uses to make decisions [4]. Generally, when an item fits within the user's preferences, they will be satisfied with the recommendation, and ultimately with their decision. Providing a more personalized approach to recommendations can improve the process immensely. For example, adaptive personalized weighting of attributes for Amazon[4] recommendations provided users with an experience in which they were able to interact with more items despite scrolling less [13].

Multiple criteria systems can provide more accurate recommendations. Users may rate an item using two criteria significantly different, while using only a single criteria might produce the same rating. For example, if a film rating system had a single-rating system, a user may rate two films as 6/10. If allowed to rate the films using story and visual effects, they may rate the two films as (3/10 and 9/10) and (9/10 and 3/10) respectively [28]. Knowing which criteria is more important to that user, and finding a way to incorporate these multiple factors into the recommendation algorithm is important in providing a more accurate recommendation.

As another way to improve the customer satisfaction within Recommender Systems, many developers have turned to explanation. This is done to help provide the user with some sort of justification as to why a specific item is being suggested. These can be done achieve numerous goals as described earlier in this paper. Probably one of the most well-known examples of these is Amazon's "Customers who bought this item also bought..." interface. This is an example of non-personalized explanation [8]. However, users tend to be more satisfied with personalized explanation than non-personalized [32].

Explanations can be used to accomplish a number of goals, as discussed previously. It appears that trust and transparency are highly important factors in satisfaction of recommendations, but decision efficiency does not have a major impact [8]. This means that users tend to value a system in which they understand the motivation behind their provided results.

Through explanation, Recommender Systems aid users in understanding provided information better and faster. As a result, these types of systems can provide results which enable users to make decisions with a higher level of confidence and satisfaction.

[4]https://www.amazon.com/

3.3 Visualization

Decision support tools like search and recommendations ease the process of parsing the increasing amount of information available on the Web. However, even if the right information is retrieved, data retrieval is only part of the mitigation of information overload. Another important factor in dealing with the problem is presenting the data in a way that the user can understand. One common way to aid users in understanding data, especially large sets, is through visualization. Visualization uses *Gestalt Principles* such as proximity, similarity, continuity, symmetry, closure, and relative size which allow users to see patterns in data [11]. By appealing to these natural human processes, visualizations can make data processing and therefore decision making a more efficient process.

There is an increasing amount of information available on the Web, but there are also an increasing amount of tools available to help understand that information. With the advent of new, free, and widely available visualization tools, even novice users can now see and understand personally relevant information [31]. Experimenting using visualization can help encourage data exploration in new ways.

3.3.1 Visualization in Search

Visual techniques can be applied to search results, helping users to parse through a wide array of information. Although humans do not perceive and process information well in linear patterns, it is still the way search engines often present their results [5]. As a result, experimentation with varying methods of presentation could provide more efficient and effective data retrieval during search on the Web.

Visualization can help users to navigate text-based search results. *ProjSnippet* introduced a way to visualize collections of text snippets, building on intuitive layouts that optimize placement of the snippets [9]. The project was developed in a way which is easy to append to search engines without affecting the response times significantly.

As search results tend to return large amounts of text, new developments in text visualization, the most common being word clouds, can help users more easily identify relevant results [31]. Deeswe et al. [5] proposed a search interface which included 3D visualization. Over half of the research participants had a positive impression of the interface. The visualization interface was also found to be more difficult to use than a standard one, but that decreased with time and use.

3.3.2 Visualization in Recommendations

Most often, recommendations are presented in some sort of list or table format. In other applications, different visual techniques are used to better communicate with the user.

Basic visual techniques are often used to help users better understand their recommendations. Some examples of this include the typical star-rating system, where users can see an item's rating or relevance based on how many stars are colored in. A popular example of this method used to denote relevance to the user is on Netflix.[5] Another application of basic visual techniques in recommendations is through the use of maps. This

[5]https://www.netflix.com

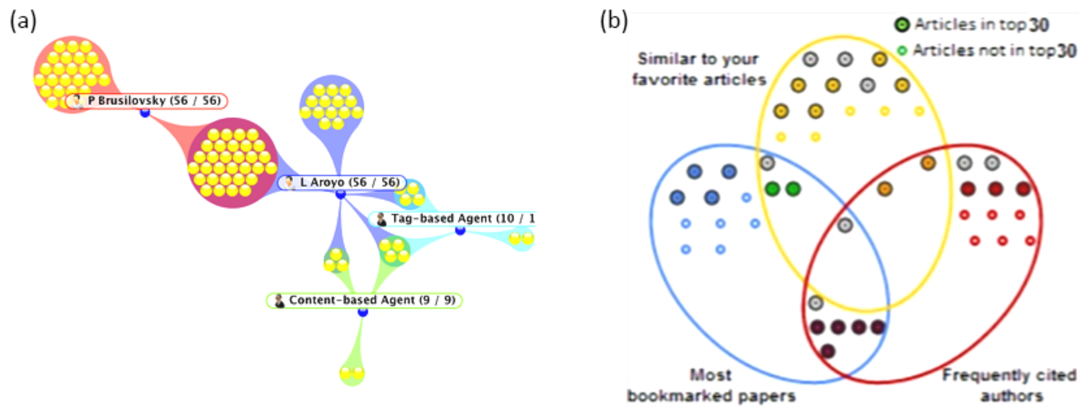


Figure 3.1: Comparison of visualizations in Recommender Systems *TalkExplorer* (a) and *SetFusion* (b)

technique can be especially useful when users search for contextual recommendations based on a specific location.

Additionally, recommendations can be displayed in a graphical format, especially popular when the suggestions are presented in groups or sets. *SetFusion* [25] uses interactive Venn diagrams to represent how recommendations match various criteria. *TalkExplorer* [34] users a similar approach with cluster diagrams, representing connections between various data nodes such as users and tags. The two visual representations can be seen in 3.1.

One trend in Recommender Systems using information visualization is to help justify results. Overall, most interactive Recommender Systems utilize visualization techniques to help users navigate through data. However, there is still a lack of systems that provide adaptive levels of visualization for different users [11].

Visualization can be used in many ways to support explanation in Recommender Systems. *TasteWeights* [2], for example, seeks to accomplish goals which allow users to update their profiles, provide feedback, and support exploration of potential scenarios through interactive visual techniques.

Another example of using visual techniques in explanation is through the use of word tag clouds. Explanations based on tag clouds are well-accepted by users, and also improve the efficiency and effectiveness of explaining recommendations [7]. By providing the results in a more novel way, users perceive a the system as higher quality.

3.4 Interactivity and User Control

Though static visualization of data can be useful, allowing user interaction provides even more power. Just as with other areas of the Web, and technology in general, visualization techniques are becoming increasingly interactive, encouraging more user exploration of, and sometimes control over, information.

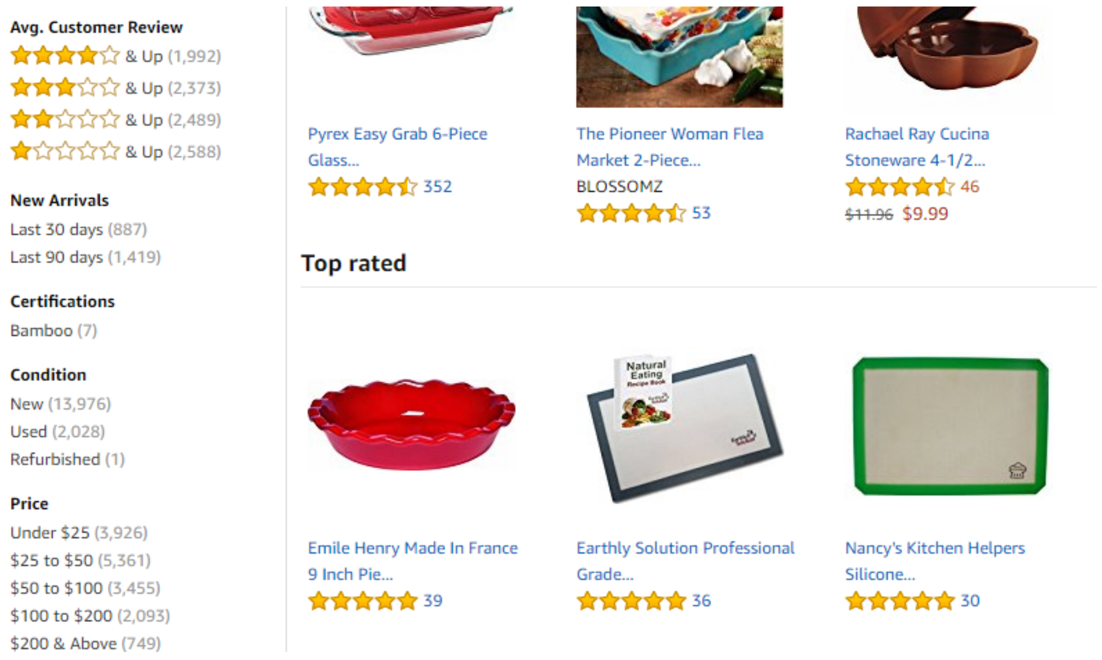


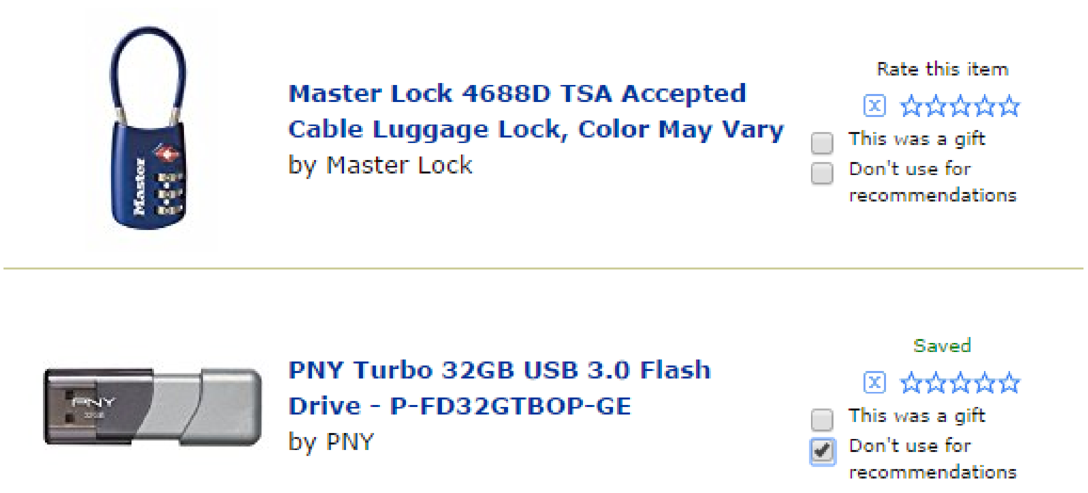
Figure 3.2: Faceted Search on Amazon

3.4.1 Interactivity and User Control in Search

One of the more basic approaches to personalization and interactivity during the search process is through the application of filters. Users can provide search queries, but choose to remove certain results based on certain criteria. Typically, users may provide multiple filters to the same search, allowing for added personalization.

A more sophisticated search technique that is similar to filtering is *Faceted Search*, also known as *Faceted Browsing*, or *Faceted Navigation*. Faceted Search provides a user-friendly alternative to keyword searching using structured information and attributes [26]. While it is similar to filtering, it offers a more defined approach. Users can narrow down a larger data set by choosing a specific predefined attribute from a list. This can then be narrowed further with attributes from either the same or a different category. Amazon's search function, shown in Figure 3.2, is a well-known example of faceted search. Here, the user can use links on the left sidebar to narrow down their search based on things such as user review score, condition, and price. In some cases, the data set can contain a large number of attributes, which means that those more relevant to the user's needs should be provided in a more accessible location [36].

One potential drawback from Faceted Search Systems, as well as filtering, is that the concrete structure means results either meet or do not meet the selected attribute. As a result, useful results based on attributes which are either optional or maintain varying levels of importance tend to be difficult to incorporate. A proposal to attack this is through weighted Faceted Search, in which users can select which attributes are mandatory and which are not [35]. From this, a ranked result set can be generated which assigns a higher score to items which meet more of the desired attributes.


Master Lock 4688D TSA Accepted Cable Luggage Lock, Color May Vary
by Master Lock
Rate this item
This was a gift
Don't use for recommendations

PNY Turbo 32GB USB 3.0 Flash Drive - P-FD32GTBOP-GE
by PNY
Saved
This was a gift
✓ Don't use for recommendations

Figure 3.3: Amazon's "Improve Your Recommendations" feature

3.4.2 Interactivity and User Control in Recommendations

Providing users the ability to maintain some level of control over their preferences in a straightforward way can be used to improve the decision process and satisfaction. Even though it requires more effort and recommendations are often objectively less accurate, users are often more satisfied when given increased control [19]. If satisfaction is the ultimate goal in a decision making system, then this means it is likely better to provide some sort of control, even if it decreases the overall quality of the suggestions.

On the most basic level, user control is prevalent among Recommender Systems in the form of profiles or preferences. Users may be asked to answer explicit questions which provide information to the system. However, allowing for additional user input to improve future recommendations is becoming more common. Amazon, for example, provides a basic feature, as displayed in Figure 3.3, which simply allows the user to choose which products purchased in the past will factor into their product recommendations and which will not.

Other studies have proposed approaches that involve the user as a more active participant in steering the algorithm. However, in exploratory searches, the user is typically not familiar with the domain, and requires some direction. One potential solution to this problem is to provide visual feedback, which helps the user predict the effects of their actions [17]. A system which provides immediate feedback helps users in completing the task.

Additionally, it appears that the best interaction methods within Recommender Systems differ depending on user characteristics. Basic and novice users tend to flock more toward methods showing top items, while experts tend to prefer more complex implicit control [18]. Finding a balance between these to create some type of hybrid which can accommodate both these things could be the key to a more successful system.

In a survey of interactive Recommender Systems [11], 14 of 24 investigated systems

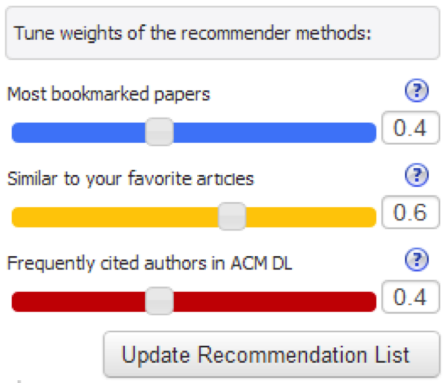

Tune weights of the recommender methods:
Most bookmarked papers
0.4
Similar to your favorite articles
0.6
Frequently cited authors in ACM DL
0.4
Update Recommendation List

Figure 3.4: Sliders controlling weighted factors for recommendation in *SetFusion*

incorporated some sort of user control as part of the recommendation process. In these systems, user control has an impact on recommendation accuracy. Most focus primarily on adjustment of user profile, though others also allow refinement of the recommendations directly as well. However, there is still a lack of systems which adapt the level of control depending on certain user characteristics.

One technique that systems utilize to improve user control is interactive visualization. Recommender Systems such as *PeerChooser*, *SmallWorlds*, and *Pharos* provide visualizations in an effort to encourage user exploration [34]. However, exploration only provides limited amounts of interactivity to influence the results themselves. *TasteWeights* [2] has proposed using visual interaction as part of a hybrid Recommender System, discovering that users felt interaction improved their recommendations. Overall, the more interaction used, the more positive the sentiment was.

In *SetFusion* [25], users are presented with an interactive Venn Diagram and set of sliders to make the recommendations more transparent, controllable, and explorable. The sliders, as shown in Figure 3.4, allow users to control the weighted influence different factors have on the recommendation list. This method showed an increased impact on user motivation, performance, and attitude. Thus, the decision-making process is vastly changed with differences in data presentation, even when the data is the same.

Another possible way to incorporate user control is to allow users to choose the actual algorithms used to generate suggestions. Hybrid systems seek to blend strengths of different approaches, attempting to best meet the needs of a specific user in a specific situation. One proposed technique to meet this requirement is through self-selection of the recommendation algorithm, rather than through explicit and implicit information extraction [6]. Though no significant feedback regarding this technique was acquired, the functionality of switching recommendation algorithms was used extensively. Having control appeared that it may promote long-term use of the system.

Many approaches are being increasingly used to integrate user control into Recommender Systems. This has shown to help increase customer satisfaction with their own decisions, as well as the systems themselves.

Chapter 4

Own Approach

Over the past years, many new systems have been developed to help users during the decision-making process on the Web. While many systems operate in a "black box" in which the user is hidden from the inner workings, recent trends have pointed in the direction of increased user control.

Additionally, presentation is becoming more and more important, as mitigating overload becomes more difficult with the overwhelming inundation of information on the Web. As a result, visualization techniques are becoming more heavily used in decision support tools.

Based on these trends, a new prototype has been developed to investigate user acceptance of a tool which focuses on immediate, interactive feedback in a visual way, while also providing the user with a higher level of control over their decision criteria input.

4.1 Problem Statement

The primary problem this approach seeks to investigate is the lack of intuitive and immediate interactive and visual tools on the Web in the decision-making domain. The prototype seeks to determine user response to a tool which gives them more control over their decisions, while presenting the information in a more visual manner.

Another issue that lacks among existing systems is a controllable way to retrieve information for individual decisions. Typically, focus is put on algorithms which generate accurate and relevant results. These results generally take input from user profiles and other gathered data. However, decision factors can change over time and differ from one decision to the next.

While these systems are helpful in making choices and retrieving relevant information, this prototype seeks to narrow the focus on the presentation interface and how the individual sessions can be improved. With additional focus on the presentation, the final step of these algorithm and calculation-heavy systems, more sophisticated systems can be developed which not only provide accurate results, but do it in a way which improves user satisfaction with their decisions. Improving the actual experience of the process for the user is likely to lead them to higher levels of satisfaction with an application and thus more likely to continue using it.

4.2 Prototype Goals

To investigate the potential for improving decision-making on the web, a prototype system was developed. In this chapter, the design of this system is explained. The overall goals of the system fall under three primary areas: user control and immediate feedback, visual presentation, and modularity and flexibility.

4.2.1 User Control with Immediate Feedback

One of the two primary areas of investigation in this paper is the effect of user control on decision support tools. Specifically, this project aims to include user control in a way that provides immediate feedback. By doing this, the user should experience an increased sense of control, and in theory increased satisfaction with their decision process. The improved sense of control for the system is focused heavily on decision criteria, allowing the user to better communicate their reasons for making the decision. With more accurate criteria selection, it should be possible to produce more relevant and effective results.

4.2.2 Visual Presentation

In addition to improving interactivity and user control, the system is designed to present the information in a simple and intuitive way. To achieve this, the data is provided in a visual way rather than through textual results. This should allow the user to be able to process the information more quickly and more easily interact with the system. Two different types of visual presentation are explored in this paper, to provide a more complete and robust evaluation and investigation of the effect of visualization on the decision process. The two visualizations: a stacked bar chart and a Venn diagram, present the data in different ways, each of which may appeal to different users. The motivation behind providing two options was to better identify whether the concept of visualization is more generally accepted, or if specific presentations are more influential on the user experience while using the prototype.

4.2.3 Modularity and Flexibility

While many domain-specific Recommendation and Search Systems are available, this system aims at producing a reusable and portable module which can be applied in multiple domains. Ideally, the aim of the project is not only to identify how users value the system itself, but also their attitudes toward user control, interactivity with immediate feedback, and presentation of information in a visual way. A modular approach also allows for extensibility in a way that different sources, algorithms, and more can be applied. This means that the results can be extrapolated to additional domains, industries, and systems, rather than applying only to this implementation.

4.3 Application Domain

One of the primary goals of the tool is to produce a modular and flexible system. However, choosing a concrete domain to provide a proof of concept was also important.

In this project, the domain of restaurant selection was chosen. The primary motivations behind this were:

- Information comes from a variety of sources and data types. Users may seek out information from various places such as Recommender Systems, expert reviews, basic location-based searches, social networks, and more.
- Decision factors can vary widely between users, but also from session to session for individuals. While a young professional may seek cheaper places for personal meals, price may not factor in their decisions as much while traveling for business.
- Users often have more than one decision factor, but put more emphasis on one or more attributes. For example, a user might seek a cheap restaurant which serves hamburgers, but also vegetarian options. However, having a vegetarian in the group makes the latter attribute more important than having hamburgers.

4.4 System Design

Architecture of the system focuses on addressing both primary goals of the study, as well as modular principles and possible future integration into other domains and systems. As mentioned earlier, the search interface developed for this prototype can be replaced or appended. Other pieces of the system aim to allow a similar flexibility. One of the ultimate goals is to provide a solution that can be put into numerous applications, regardless of the domain or the type of information which must be retrieved.

Using the example of the restaurant domain, users may look to many different places to help during their decision process. Recommender Systems and venue searches such as Foursquare[1], TripAdvisor[2], or Google Maps[3] can provide users with criteria-based lists of appropriate options. Additionally, some may turn to sources such as lists of top restaurants in a city written by an expert in the area, or to their friends for a more social approach.

To accommodate this, the system allows for a high level of control in the first step of the decision process: data retrieval. The core system then primarily dedicates itself to the aggregation of data sets, user control over attribute weighting, and the visual display interface. This approach is visualized in Figure 4.1.

4.4.1 Data Retrieval Interface

Data can come in many shapes and sizes, from many places. Ultimately, the system is designed in a way which could integrate any number of data types and sources in the future. By putting the focus on the attribute weighting and result presentation, where those attributes and data sets actually come from becomes less important.

The first step of the designed system, as in many decision support systems, is a data retrieval interface. An example interface, utilizing the Foursquare Venues API[4] is included for demonstration and testing purposes. However, additional or alternative

[1]https://foursquare.com/
[2]https://www.tripadvisor.com/
[3]https://www.google.com/maps/
[4]https://developer.foursquare.com/overview/venues.html

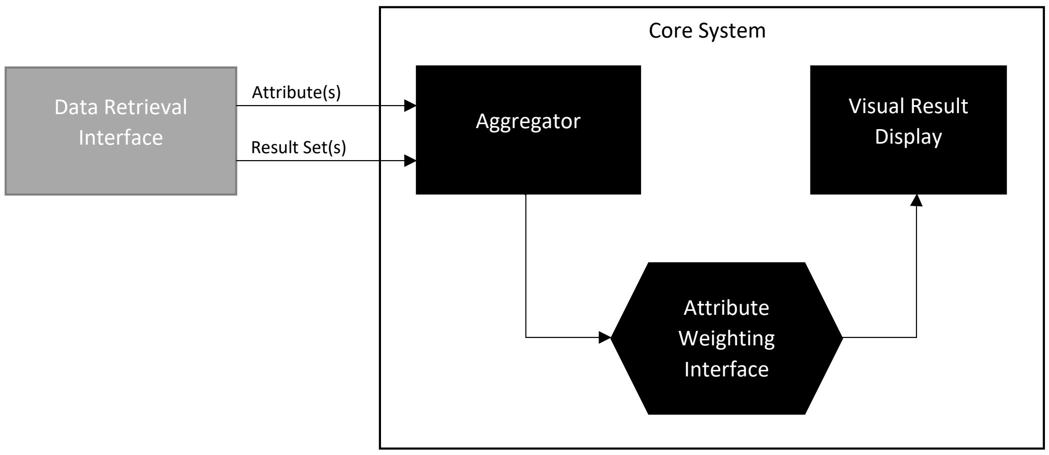


Figure 4.1: General overview of system architecture

search interfaces can be substituted in place of the implemented component. This helps to address the goal to provide an extensible and modular application. More information about how the search components work can be found later within Chapter 5.

For this paper, a search component has been implemented within the restaurant domain. Users are asked to search up to four criteria which apply to their current decision. Searches were limited to four attributes, as additional ones generally cause more clutter and often cloud the results more than smaller searches. Each search is comprised of a query term as well as a location. Once the location is identified in the initial query, it cannot be changed without beginning the process again. Naturally, trying to integrate queries for restaurants within four different geographic areas would be of little use.

4.4.2 Aggregator

After the appropriate data retrieval process is complete, the system takes an input consisting of an attribute and a result set. It then aggregates the data and scores them according to the defined algorithm. By default, the score will be calculated based on each item's ranking within the results list. Each venue returned by any query receives a ranking value for each query. Then, a total score for the venue is calculated based on these individual scores and the current weight of each attribute. Additionally, an interface is provided which allows the data retrieval adapters to define a scoring function for each item.

As the system focuses especially on multiple-attribute decision processes, the aggregator can receive multiple inputs in this format. When a new data set is received, the overall data set is aggregated according to the defined scoring function. It is possible to include a custom scoring function in the adapter component, but scores are calculated simply by ranking within the results list by default. For the default scoring, the maximum possible score for a venue is 100, which would mean that it lies first in the ranking for all existing queries. Once the new list of results is identified, they can be

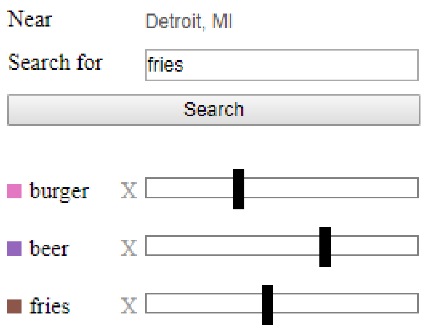


Figure 4.2: Slider interface for attribute weighting

passed along to the interface for attribute weighting and visual display.

4.4.3 Attribute Weighting Interface

Aggregation is done as a preliminary step to the final result display. Though it is important to establish a unified list that includes all attributes, the core system seeks to present this aggregated data in a new way. As this system seeks to introduce an increased amount of control and interactivity to the search process, attribute sliders, as shown in Figure 4.2, are provided for each attribute. This allows the user to identify which attributes of the set should influence the results most and least.

4.4.4 Visual Result Display

The final user interface piece of the system is the visual results display. Often, results are presented in a way which is not efficient to parse or is difficult to understand. The goal of this interface is to use visual techniques to help users come to a quicker and more effective decision.

To provide a more thorough investigation into visual techniques, the system can display results in two different ways. Ideally, the visual techniques would aim to communicate better to users which results best match their criteria overall, as well as how each individual attribute factors into that. Additionally, interactive and immediate feedback during the attribute weighting process encourages the user to explore the interface more.

The first technique displays the query results in the form of a stacked bar chart, shown in Figure 4.3. Each bar represents an individual restaurant, which each colored portion corresponding to a specific query. The aggregator component of the system calculates overall scores based on each attribute, which determines the width of that section for each bar. Once these scores are calculated, the restaurants holding the top 10 overall scores are displayed in the stacked bar chart, ordered from top to bottom.

The second technique for displaying results is through a Venn diagram, shown in Figure 4.4. Each of the attributes is represented by a large circle area, which overlap depending on how many restaurants fit into one or more of the attributes. Each restaurant is depicted using a smaller circle inside of the appropriate region. To be able to easily

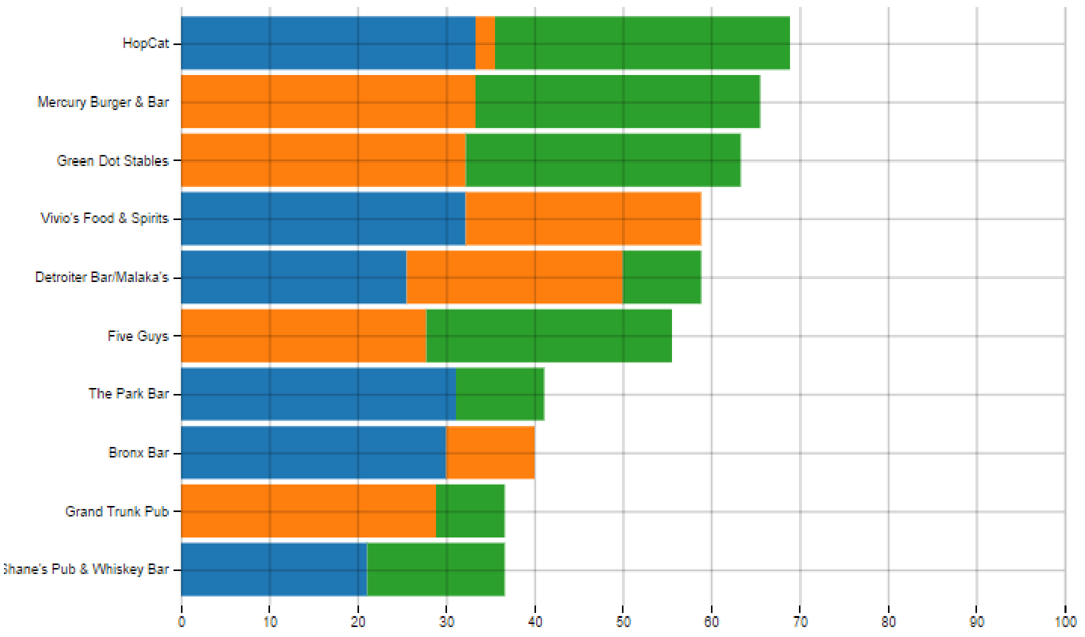


Figure 4.3: Visual display technique 1: stacked bar chart

identify the restaurants which fit within each region, all restaurant nodes are depicted at the same size and color. However, to differentiate how well the results match all queries best, the circles opacity is adjusted. Nodes which have a lower score will have more transparency, and those with higher overall scores will appear darker. Therefore, users can obtain a larger amount of information from the visualization. They can immediately identify which attributes the restaurant matches, as well as the overall match level when accounting for all criteria.

It may be possible that one restaurant has a higher score than another, even though it does not match as many criteria. For example, one restaurant that matches two of three search terms very well could obtain a much higher score than another which only slightly matches all three terms. To reduce the amount of unimportant information and clutter, each region is limited to a maximum of ten restaurant nodes. Additionally, any node which does not reach a score of at least 20 out of 100 is not displayed. This prevents restaurants which do not really match the criteria well from being shown to the user and adding irrelevant overload which could distract from the simplicity of the interface.

To encourage exploration of the results and provide a more accurate display of information, the size of the regions is adjusted whenever an attribute weight is changed. This affects the size of the larger attribute regions. Additionally, as more weight is added or removed from an attribute, the circles in that region will appear darker or lighter respectively, as adjusting the weight of that attribute would also adjust the overall score for each node which matches it.

In the stacked bar chart technique, the overall scores and restaurant names are already easily readable. However, this information is not quite so obvious in the Venn diagram visualization. To combat this, a small informational tooltip containing the

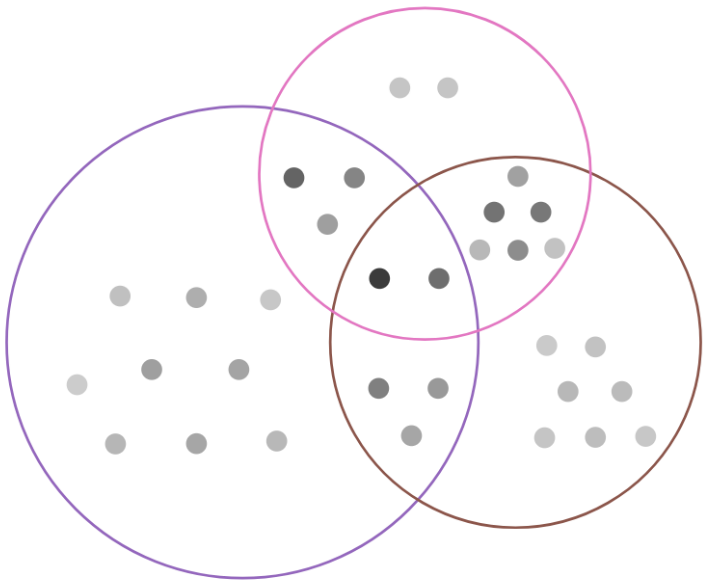


Figure 4.4: Visual display technique 2: Venn diagram

restaurant name as well as the overall score for the node, is displayed when the restaurant node is clicked, as shown in Figure 4.5. By default, the tooltip for the restaurant node with the highest overall score is displayed. This allows the user to obtain more precise information about each restaurant in addition to the more general overview which is provided by the visualization. Looking forward to additional development, this is a location where additional information could also be added, such as a link to the restaurant website, location information, or menus. For simplicity, only the total score is displayed in this prototype.

In addition to visualization, a key component of this system is the amount of control and interaction a user has over the results. When adjusting the slider, the visualizations are updated immediately to represent the change in criteria weight.

In the first technique, the bar sections representing an attribute with higher weight will show up larger and contribute more to the overall width of the bar. A comparison of this change can be seen in Figure 4.6. The left image represents the display when all attributes have equal weighting, and the right in which the attribute “beer” (orange) has been increased.

In the second technique, a similar response is triggered when the weights are adjusted. The opacity of the restaurant nodes are adjusted according to the new weighted total score, and the overall region size changes as well. Figure 4.7 shows a side-by-side comparison with varied weights. The left image represents an equally-distributed weighting, and the right shows the results where the weight for attribute “beer” (orange) has been increased.

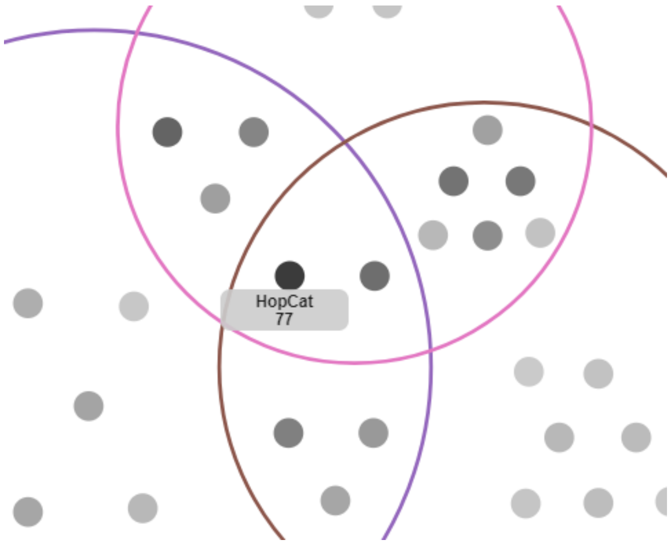


Figure 4.5: Tooltips for Venn diagram visualization, providing additional information

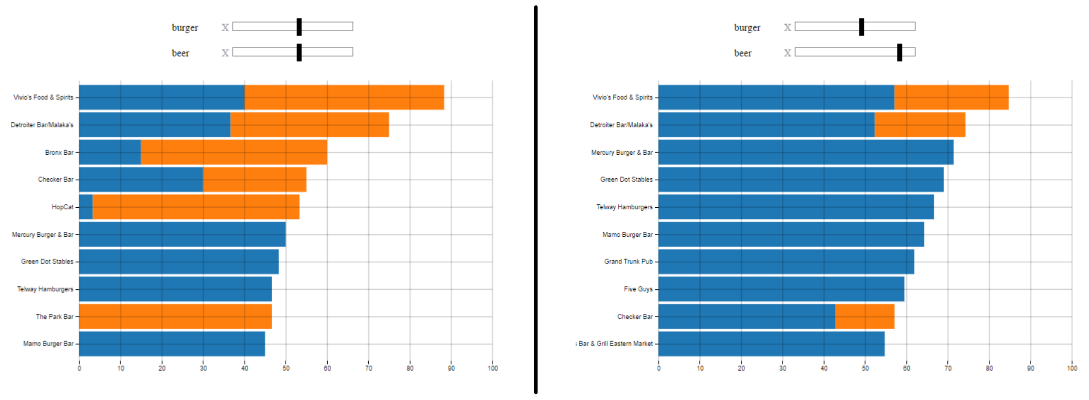


Figure 4.6: Stacked Bar Chart: Adjustment based on attribute weight

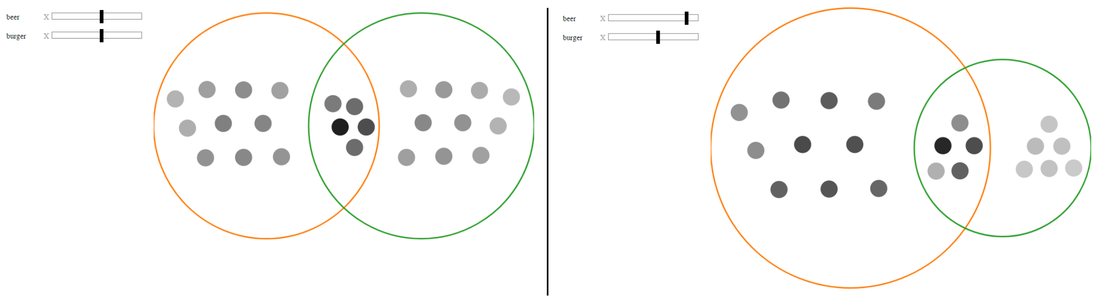


Figure 4.7: Venn Diagram: Adjustment based on attribute weight

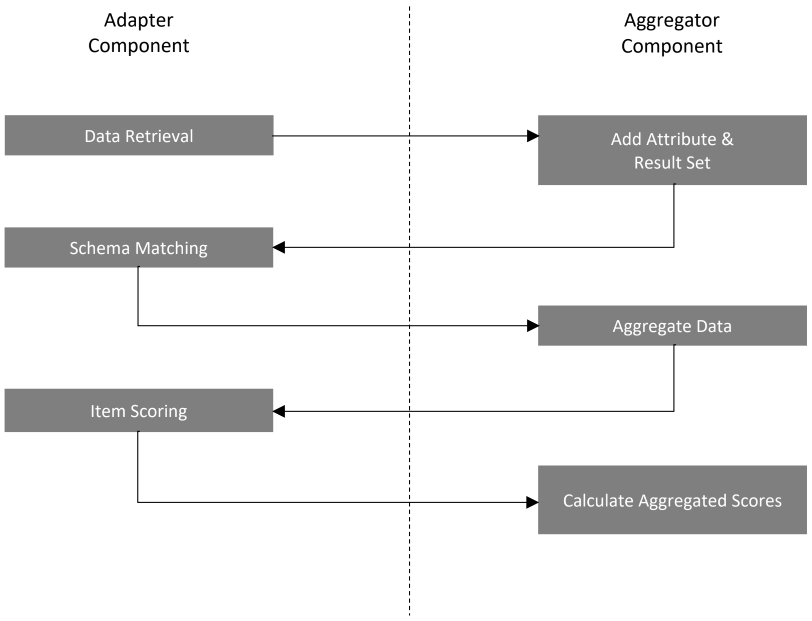


Figure 4.8: Architecture for system extension using adapter functions

4.5 System Extension and Adaptation

The core system of this application is designed with future extension in mind. Because of the separation between the core functionality and the data retrieval process, it can be included into any number of applications across numerous domains. The possibility for extension seeks to cover a few specific areas outside the core functionality. Ultimately, the system utilizes adapter functions which can pull information from the parent application to allow for additional compatibility. The way in which this process works is displayed in Figure 4.8.

4.5.1 Data Retrieval

As described earlier, one of the core goals of the system design is to allow for additional possibilities when it comes to data retrieval. To achieve this, the system allows for custom components to be created which actually provide the attributes and queries to the aggregator. This means that developers in the future can create retrieval components which can distribute results accordingly to the system. Though the result sets in the prototype developed comes from a Foursquare-API-based component, this data could potentially be retrieved from other APIs, static ranked lists, or other sources.

4.5.2 Schema Matching

One primary challenge of data integration across multiple sources is in schema matching. Using the restaurant domain as an example, different APIs would almost certainly return different unique identifiers. To handle this, it is important to use matching techniques to determine when two items are actually the same result. Though the current implementation uses only one data source, a function can be defined to check whether items are equivalent or not. As an example, identifying information such as name and address could be combined to determine whether two restaurant results are the same location or not. Thus, multiple source retrieval interfaces can be implemented and input into the aggregator.

4.5.3 Item Scoring

Another way that the system looks to future modification is through item scoring. By default, results are scored for each attribute based on their ranking within an ordered list. However, it is possible to define a scoring function within the parent application. With this functionality, developers have the ability to influence how the results are aggregated without having to modify the actual aggregation component.

Chapter 5

Implementation

The previous chapter discussed the general architecture and interface design of the system. This chapter aims to provide more insight into the technical implementation of the prototype. This includes the libraries which support the core of the system, as well as the general code architecture and structure. Additionally, the functionality of various classes, methods, and components is revealed.

5.1 System Architecture

The prototype system is built exclusively using front-end technologies. By encompassing everything within JavaScript modules, the system becomes more modular and extensible. By doing this, the core functionality can be implemented into any application using only a single imported module. The majority of code for this prototype is written in *ECMAScript 2015*[1] (ES2015), sometimes also known as *ECMAScript 6* (ES6). *HTML 5.1*[2] and *CSS*[3] are used for basic structure and styling respectively.

5.1.1 Frameworks and Libraries

As in most systems, a number of existing frameworks and libraries have been utilized in the development of this prototype. These allow the focus of development to center around the core components themselves. The following frameworks and libraries were used in the core development of the system.

React: Due to the nature of the system, it was important to select a robust single-page-application framework with a high emphasis on interactivity and immediate feedback. As a result, Facebook's *React* framework[4] was chosen. React is a framework which focuses on efficiently updating and rendering components when data changes. This is perfect for a visual, interactive system because it allows the user to make changes which impact the presentation and see them immediately. The component-based nature also allows an

[1]https://www.ecma-international.org/ecma-262/6.0/
[2]https://www.w3.org/TR/html51/
[3]https://www.w3.org/TR/CSS/
[4]https://facebook.github.io/react/

easier separation of the different aspects of the system. One component can handle the input portion, another the attribute weighting, and another the visual display of results. This is also useful for modularity, because the input module can consist of any number of input techniques, while the weighting and visual display portions remain unchanged.

D3.js: Due to the heavy focus on visual display for this system, a robust library for JavaScript-based visualization is an important piece. *D3.js*[5] offers the most extensive and flexible and well-supported visual framework available. It provides the most customizable development experience for creating visual displays on the Web.

React-D3: To utilize the D3 library more effectively within the React framework, the *React-D3* library[6] was chosen. This library provides an wide range of data visualization possibilities utilizing the D3 core, but presented in importable React components. For the purpose of the stacked bar chart data display, the `BarStackHorizontalChart` component is used. More detailed implementation of this is discussed later in the chapter.

D3-Venn: For creating the Venn diagram data display, a new component based on the *D3-Venn* plugin[7] was used. Unfortunately, the library itself does not utilize React components, so the library had to be ported into React for use in the system. More detailed implementation of this is discussed later in the chapter.

5.2 System Components

As described in the Chapter 4, the data retrieval aspect of the system exists in an external component which can be provided into the core system. This core system is made up of three main pieces: an aggregator, an attribute weighting interface, and a visual result display. In addition to these three core pieces, a few other smaller components are vital in the implementation of the final system. The hierarchical structure of components within the system can be seen in Figure 5.1. The black boxes represent core system components, the lighter gray adapter components, and the white boxes helper or display components.

5.2.1 SearchAggregator

The largest and most important piece of the system is in the aggregator component. The React class component `SearchAggregator` is responsible for handling input of data sets, combining the data together, handling changes to the attribute weighting, and passing all of this information to the desired visual display component.

Each time a new query is passed to the `SearchAggregator` via the `newQuery` property, it calls the React default `componentWillReceiveProps` method, which immediately validates the query. If the query has already been added, or if there are already four queries in the system, it will return false and alert the user with a message describing the respective problem. For each search object in the search property, the object is

[5]https://d3js.org/
[6]http://www.reactd3.org/
[7]https://github.com/christophe-g/d3-venn

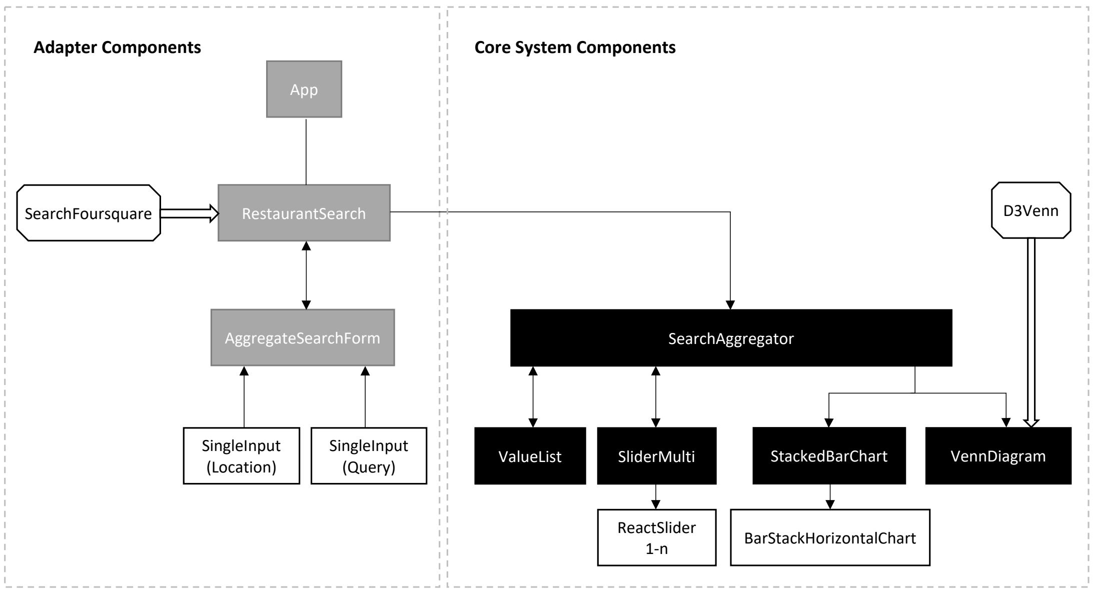


Figure 5.1: System component architecture hierarchy

iterated to retrieve search results. The code contained for iteration of the search object is found below:

```
let resultCount = 0;
for(let search of this.searches) {
  search.getResults(nextProps.newQuery).then((result) => {
    resultCount++;
    this.handleResults(result, nextProps.newQuery.query, search.idFunction, search.
     nameFunction);
    if(resultCount === this.searches.length) {
      this.queries.push({
        name: nextProps.newQuery.query,
        weight: 5
      });
      this.drawChart();
    }
  });
}
```

In each iteration, the `getResults` method is called, passing the new query as an attribute. Once the promise is returned via the search's `getResults` method, the results are sent to the internal `handleResults` method. The method takes in the result set and the query string, as well as an ID function and name function from the search adapter as input.

In the `handleResults` method, the aggregator iterates through each result to include it into the internal `resultItems` property. After processing the results, the `resultItems` array is updated to represent the venues within the overall results. Each result object within the array contains the following structure:

```
{
  'id': 'venue_id',
  'name': 'venue_name',
  'queryRanks': {
    'query_1': 100,
    'query_2': 50
  }
};
```

First, the method checks whether the object is already in the array, by using the search adapter object's `idFunction` method. In the case of this prototype, the Foursquare API Venue ID is simply returned, though this function could be used for schema matching between multiple sources. Based on this comparison, if the venue is not already included, a new object is added to the array. The ID is determined using the same method, and the name of the node is retrieved via the adapter's `nameFunction` method. This allows customization of the display information depending on the data source. Lastly, the `queryRanks` property contains one entry to represent the new query, with the key being the query name and the value being the venue score for that query. If the venue already exists, a new entry into the `queryRanks` attribute is added, and the other attributes remain.

Using a counter variable, once it is determined that all results are handled, the new query with a default weight of 6 is pushed to the `queries` array, and the `drawChart` method is called.

To retrieve the proper data for display, the query names and weights, as well as the `chartData` to be passed into the visualization are retrieved. Query names and weighting values are retrieved using the `getQueries` method, and the `chartData` is retrieved via `getTopVenues`. The code for this method is shown below:

```
getTopVenues() {
  let allVenues = this.resultItems.slice();
  let totalWeight = 0;
  for(let q of this.queries) {
    totalWeight += q.weight;
  }

  for(let venue of allVenues) {
    let score = 0;
    let ranks = venue.queryRanks;

    for(let query of this.queries) {
      if(!ranks[query.name]) {
        venue[query.name] = 0;
      } else {
        let queryWeight = query.weight / totalWeight;
        let weightedScore = 100 * queryWeight * ranks[query.name];
        venue[query.name] = Math.round(weightedScore);
        score += weightedScore;
      }
    }

    score = Math.round(score);
    if(score > 100) score = 100;

    venue.totalScore = score;
  }
    allVenues.sort(function(a,b) {
      return b.totalScore - a.totalScore;
    });

  return allVenues;
}
```

Here, the overall score values are calculated for each venue, based on its `queryRanks` values, as well as the current weight of each query. Each time a new search is performed, or the attribute weighting changes, this score is recalculated. Once all scores are recalculated, the method returns the full list of venues, sorted by descending score.

After this calculation is completed, the React state is updated, triggering the visualization component to update with the newly set properties.

5.2.2 SliderMulti

One component which is a child of the `SearchAggregator` component is `SliderMulti`. This component displays a slider interface for each added attribute allowing the user to adjust attribute weighting. Whenever the user modifies one of the slider components, the `handleWeightChange` method within the `SearchAggregator` is triggered, receiving the values of each slider from the `SliderMulti` component. Additionally, the sliders will only display if there are at least two query attributes in the component properties.

5.2.3 ReactSlider

The `SliderMulti` component contains n child `ReactSlider` components, where n is the number of attributes in the current state of `SearchAggregator`. These components are imported from the *react-slider* plugin[8]. When any of the child sliders are modified, the parent component is triggered as changed via the `weightChange` method property, allowing it to pass the information back to the aggregator.

5.2.4 ValueList

The `ValueList` component is a component which displays the current attribute terms. For each new query, the value is displayed via this component. Additionally, users have the ability to remove existing terms from the results. When the remove button is clicked, a callback in the parent `SearchAggregator` component is triggered via the `removeFunc` property.

5.2.5 StackedBarChart

One of the two visualization types used to display the query results in the system is a stacked bar chart. The `StackedBarChart` component receives properties as input from the `SearchAggregator` component and displays it. The main functional purpose of the component is to define the data which should be used for the x and y axes on the chart. To display the data properly, the `chartSeries` and `chartData` properties must be passed into the `BarStackHorizontalChart` component in the correct format.

[8]https://github.com/mpowaga/react-slider

The `chartSeries` property should contain an array of objects, each representing one query, in the following format:

```
{
  'field': 'query_1',
  'name': 'query_1'
}
```

The `chartData` property should contain attributes for each series passed via the `chartSeries` property as the key, and the score value for that query as the value, as well as a name for the data set. In the case of this application, the total value of the scores should add up to no more than 100. These scores are calculated according to each venue's score as well as the weighting of each query attribute. Including additional attributes does not cause problems, as long as the required properties exist in each object. The format of the data should look as follows:

```
{
  'name': 'venue_1',
  'query_1': 25,
  'query_2': 25,
  ...,
  'query_n': 25
}
```

Because the `SearchAggregator` results object is not formatted in this way, the component contains a method for adapting the data passed through the properties. When new data is passed to the `StackedBarChart` component, the `render` method is automatically called to generate the new stacked bar chart. However, since the series and data properties are not properly formatted to be passed to the display component, the `getSeries` and `getData` methods are called before rendering the component. Within these methods, the existing series and data structures are converted to comply with the `BarStackHorizontalChart` formats.

In addition to formatting the data properly for display, the component also defines other visual properties of the chart, such as chart size and how the axes should be displayed.

5.2.6 BarStackHorizontalChart

The `BarStackHorizontalChart` component is a component which is imported from the React-D3 library. This component takes data and visual settings as property inputs and outputs the information as a stacked bar chart. This library is the reason why the data must be structured in a specific way by the `StackedBarChart` component.

5.2.7 VennDiagram

The second visualization type used to display query results is the Venn diagram. The `VennDiagram` component receives properties as input from the `SearchAggregator` component and displays it. The purpose of `VennDiagram` is to take the data to be displayed as input, format it accordingly, and generate the chart itself. The `chartData` property should contain an array of objects which identify the venue's name and the queries which it belongs to. Also important is the final calculated score calculated during aggregation. Additional properties within each object do not cause a problem, just as in the StackedBarChart component. For the chart to be displayed correctly, it must be in the format displayed below:

```
[
  {
    'name':'venue_1',
    'set':['query_1'],
    'totalScore':45
  },
  {
    'name':'venue_1',
    'set':['query_2']},
    'totalScore':40
  {
    'name':'venue_2',
    'set':['query_1','query_2'],
    'totalScore':100
  },
  ...
]
```

When the data or queries of the component are updated, a method createChart is called to generate the new Venn diagram. This method converts the input parameters into the proper format as described above for generating the chart. It is important for the data to be formatted into the proper structure, as the D3Venn component utilizes a specific structure to properly display the chart.

Additionally handled within the createChart method is creation of the tooltips and unification of the colors within the chart with the ValueList component, which displays the current queries. A small square is added to the list to indicate which large circle region in the Venn diagram corresponds to which attribute.

A method getTooltip is used to return the proper information to be displayed within the tooltips. By default, the tooltip for the highest-scoring node is displayed during the generation of the chart. It can be removed by clicking on a different node to display that venue's tooltip, or clicking on another portion of the interface to remove the tooltip entirely.

5.2.8 D3Venn

The D3Venn component was created by porting the existing d3-venn library into an ES6 class. To create the Venn diagram, the D3Venn class must be injected into the VennDiagram component. The logic for actually displaying the diagram comes from D3Venn. As a result, the data structure described above must be used to generate the chart.

5.3 Adapter Components

While the core components of the system do the majority of the work, several components were created as part of the prototype which serve as adapters. These components could be replaced with different ones depending on the application and its domain. With this structure in place, the core components can be utilized by an number of types of applications.

5.3.1 SearchFoursquare

For the purpose of this prototype, the data is retrieved via the Foursquare Venues API. The `SearchFoursquare` class defines the necessary configurations, as well as providing an interface to connect to the API. This is then imported into `RestaurantSearch` to be used in the data retrieval process. By importing the class here, rather than within the core `SearchAggregator`, an increased level of modularity and flexibility is maintained.

In the future, even within this prototype in the restaurant domain, the search aspect could be further extended to include similar components for data retrieval. For example, a corresponding TripAdvisor or Google Maps component could be added. Regardless of the domain or application, to function within the core `SearchAggregator`, these components must include the following methods:

- `getResults(newQuery)`: This method is responsible for retrieving the desired data. In the case of this restaurant component, it accesses the Foursquare Venues API `explore` function, and returns the list of restaurants based on the query input. The property `newQuery` should contain attributes `loc` for the location string and `query` for the search term.
- `idFunction(item)`: This method is used for the purpose of data aggregation. While combining multiple data sets, providing an option for schema matching is important. This method takes the item (in this case, a restaurant object) as input and returns an object back which uniquely identifies that item. Because this prototype only utilizes a single API, the Foursquare venue ID is used. However, more complex information could be used. As an example, with multiple different restaurant-based APIs, the identifying features such as the restaurant name and address might be used.
- `nameFunction(item)`: This method is used for display purposes. To accommodate different types of data structures returned from different access methods, an adapter is necessary to show the proper label for an item. In this case, the Foursquare venue name is relayed.

React Native Foursquare API: The actual data retrieval and connection to the Foursquare Venues API is done through an imported component called react-native-foursquare-api.[9] This allows for a simpler setup of the configuration as well as callback functions after the data is retrieved.

5.3.2 Presentation and Wrapper Components

To support the `SearchFoursquare` adapter component, a few additional small presentation and wrapper components are included in the adapter portion of the interface. This allows the data retrieval process to be supported and displayed to the user, as well as increased flexibility.

App: The `App` component simply serves as a top-level application component. It initializes the React application and designates the `RestaurantSearch` component in its render method.

[9]https://github.com/lwhiteley/react-native-foursquare-api

RestaurantSearch: `RestaurantSearch` is the primary adapter component which houses much of the customized logic for this particular domain. Here, various methods and objects are available which are responsible for both retrieving data and the potential for processing. Its primary purpose is to define the various components which do the data retrieval and to pass along queries into the core system via `SearchAggregator`.

AggregateSearchForm: An interface for the actual input of query information is necessary for proper data retrieval. In the `render` method within `RestaurantSearch`, the `AggregateSearchForm` component serves this purpose. The form consists of two fields: location and query. Once the user has entered both fields and clicked search, the form submits and returns via the `handleSubmit` method to `RestaurantSearch` for information processing and retrieval of the necessary data. The `AggregateSearchForm` does not need to be aware of the various search components, nor of the core system itself. It is simply a helper component which retrieves the user's input.

SingleInput: A basic display component is used for the input fields themselves. Two instances of `SingleInput` comprise the `AggregateSearchForm`. This component simply receives properties for various HTML attributes and renders an element containing a label and text input.

5.4 Build and Deployment

To maintain a more structural approach, various build and packing tools are used. With this, a better separation of components, libraries, and other application pieces can be maintained.

5.4.1 Node Package Manager

For the purpose of library inclusion and building the application, Node Package Manager[10] (NPM) is used. The `package.json` file maintains all configuration items related to NPM. The primary pieces of functionality here are in the `scripts`, `babel`, `dependencies`, and `devDependencies` attributes. As these libraries are not included in the source code for the application, they can be installed using NPM prior to building and compiling, using:

```
npm install
```

The `scripts` component defines the build commands for the application. Based on these definitions, the application can be built using:

```
npm run build
```

A more compressed and portable version can also be built. This version includes minification and compression, ideal for production distributions. This build of the application can be built using:

```
npm run buildp
```

[10] https://www.npmjs.com/

A majority of the external libraries used in this prototype are included via NPM. Some of these libraries are core elements in the JavaScript code run by the browser. These runtime dependencies, along with their versions, are shown below, as they are defined in `package.json`:

```
"dependencies": {
  "d3": "^4.7.3",
  "react": "^15.4.2",
  "react-d3-basic": "^1.6.11",
  "react-dom": "^15.4.2",
  "react-native-foursquare-api": "0.0.4",
  "react-slider": "^0.7.0"
}
```

5.4.2 Webpack and Babel

To bundle the modules and compile the code for use within the browser, numerous development dependencies are necessary. These libraries are not executed by the browser at runtime, but instead are used to build the application code prior to deployment. These dependencies, along with their versions, are shown as defined in `package.json` below:

```
"devDependencies": {
  "babel-core": "^6.21.0",
  "babel-loader": "^6.2.10",
  "babel-preset-env": "^1.1.6",
  "babel-preset-es2015": "^6.18.0",
  "babel-preset-react": "^6.16.0",
  "webpack": "^2.2.0-rc.3"
}
```

For module bundling, the application uses *Webpack*[11]. Configuration for the Webpack build is found in `webpack.config.js`, as seen below:

```
module.exports = {
  entry: './src/client/app/index.js',
  output: {
    filename: 'app.js',
    path: './src/client/public'
  },
  module: {
    loaders: [
      {
        test: /\textbackslash.js\textdollar/,
        exclude: /node_modules/,
        loader: 'babel-loader'
      }
    ],
  }
}
```

This defines the entry point for the application, in which the source code lies, as well as the output destination path and filename for the bundled code. It also identifies which code to ignore and how it should be compiled in the loaders property.

[11]https://webpack.github.io/

To compile the code from ES2015 to browser-compatible JavaScript, the *Babel* loader[12] is used. Various presets are included to define how the compiling should be done. The configuration, as displayed below, is housed within package.json:

```
"babel": {
  "presets": [
    "react",
    "es2015",
    "env"
  ]
}
```

This configuration tells Babel that the input code is written using React, as well as ECMAScript 2015. The `env` preset can be used to define specific target environments. However, with the basic configuration included here, the code is compiled using the default options.

5.4.3 File Structure

For easier maintenance, and to accommodate the various tools used in building the application, the application is structured in a particular manner. All included NPM-based libraries are installed via the tool into the `/node_modules` directory. The markup for generating the page shell is located at `/src/client/index.html`. Due to the basic nature of the application, the styling is also included in the HTML file within the `<head>` tag. All source component files are included within `/src/client/app` directory. As shown in the Webpack configuration above, the main entry point for the application is included at `/src/client/app/index.js`, and the compiled JavaScript code is distributed to `/src/client/public/app.js`. This is the JavaScript code that is actually read by the user's browser.

[12]https://babeljs.io/

Chapter 6

Evaluation

When developing a prototype, it is important to evaluate whether the system is able to reach its defined goals. In applications where user experience and satisfaction is a primary exploration, this process becomes more vital. The application prototype developed for this thesis investigates the impact of visualization and control on user satisfaction and decision-making using Web tools. To properly determine the effects of these factors on the user experience, an evaluation was performed. This chapter discusses the design, implementation, and results of a user study with the intention of determining both the effectiveness of the system itself, as well as the general impact of visualization and user control on Web-based user decisions.

6.1 Evaluation Goals

The purpose of evaluation for this prototype is two-fold. First, it seeks to investigate the usability and effectiveness of the system itself. However, the evaluation also aims at determining user attitudes toward visualization, control, and interactivity within Web-based decision support systems on a more theoretical level. By providing a testable interface to the subjects of the study, they are able to better identify their attitudes toward these characteristics. Though the test itself reveals information specifically related to the actual prototype, it can also provide insight into general effects of the different attributes.

The evaluation of the prototype seeks primarily to answer the following questions:

- How does presenting results and items in a visual way affect a user's decision process? How does it effect the user's satisfaction with results?
- How does providing interactivity with control over results change the user experience and satisfaction during the decision process? How does it effect the user satisfaction with respect to the results themselves?
- Do users feel that increased control and/or do visual display techniques improve (or detract from) the decision process?
- Does the system tested match these general attitudes?
- Could the prototype be changed in a way which better utilizes these techniques and improves the user experience of decision making?

6.2 User Study

As this prototype primarily aims to investigate the effects on user experience, the most logical evaluation is through a user study. To determine user attitudes on the system, as well as general opinions of visualization and user control during the decision process, a study was designed to simulate ways in which such a tool would normally be used. As a result, the study was conducted on "normal" users, with no particular attention to domain knowledge or specialization. With this strategy, the system achieves a better sample of real-world application.

6.2.1 Design

The user study was conducted in three parts in a single setting. Each user was required to complete three specific scenarios based on real use cases for a system such as the prototype. Prior to beginning the tasks, the user was given a short written overview explaining how the system works. Users were encouraged to ask for any help while using the system if necessary. After the short introduction, links were provided to each interface, as well as the survey to complete after both prototypes were tested.

To control for bias, half of the users were presented first with the Stacked Bar Chart visualization technique, and the other half were presented first with the Venn Diagram visualization technique. Regardless of which visual display was completed first, all participants completed the defined scenarios using both interfaces. This allowed for the users to compare the different visualizations, while also getting a more general opinion of visualization techniques. The defined scenarios were performed in the same order by all participants, as any learning curve or familiarity factors should not be affected by the type of scenario. The first two scenarios provided specific constraints and suggestions to mimic potential realistic use cases. The final of the three scenarios was designed specifically to allow freedom of choice, leaving much of the exploration and decisions to the user. This allowed analysis of typical use scenarios, as well as an opportunity to see how the systems would be used with less specific direction. The freedom of choice was delivered as the final scenario to allow the users to gain some familiarity with the system before having extensive liberty.

6.2.2 Survey

The survey in this study was devised to allow for general attitudes about user control and visualization in the decision making process, as well as to identify user opinion on both prototype interfaces. After participants completed the study scenarios in each interface, they were linked to a survey created using *Google Forms*.[1]

Survey Structure The study was divided into four sections. The first section asked participants about their general attitude toward the prototype. The second and third sections requested users to rate their impression of each type of visualization individually. Participants were presented with the two sections in the same order in which they evaluated each interface. Therefore, users who completed the scenarios with the Stacked

[1] https://docs.google.com/forms/

Bar Chart visualization type first were also presented with the survey section for the Stacked Bar Chart first. The fourth and final section focused on general attitudes about the concept as well as demographic information. Lastly, a few questions regarding user habits in terms of Web-based decision making were presented. These questions sought to explore participants past technical knowledge and experience with similar systems.

Data Collection The first three sections all focused primarily on obtaining quantitative data, using a *Likert Scale* rating system [22] which allows the user to rate their impression of various attributes with values ranging from 1 (negative) to 5 (positive). The fourth section collected definite demographic information including nationality, age, and gender. Classifiable data was used to identify the knowledge and experience of users with decision systems and technology in general. Qualitative questions allowed the user to express attitudes and reactions from the study in an open-ended format.

6.2.3 Participants

A total of 23 people participated in the study, 11 beginning with the Stacked Bar Chart visualization, and the remaining 12 evaluating the Venn Diagram interface first.

Demographics To better accomplish a representative sample of users for the study, one aim in selecting participants was to find a diverse group. Out of the 23 participants, 12 identified as male, 10 as female, and one chose not to disclose gender. The study sampled users from at least seven countries around the world, with a plurality of 10 coming from the United States of America. Six of the participants elected not to disclose their nationality. The average age of participants was 34.2 years old, ranging from 22 to 62.

Domain Knowledge Though no specific effort was made to identify any amounts of domain familiarity or technical knowledge prior to the study, information was collected about these areas. A large majority (19 of 23 participants) users indicated that they use computers more than four hours per day. All users responded that they use computers at least once per day. Additionally, users were asked how often they use decision-making tools in the restaurant domain, as well as decision-making tools in other domains. Only 13.0% (3 of 23 participants) indicated using each type of tool at least once per week. However, only two users claimed to never use restaurant-related decision tools. All users indicated that they use decision making tools in some domain on the Web at least once per month.

Decision Time The survey also sought to identify how much time participants typically spend while deciding where to eat in restaurants. 17.4% (4 of 23 participants) said they spend more than 30 minutes deciding where to eat, and only 8.7% (2 of 23) said they spend less than five minutes deciding. Therefore, most users usually spend between 5 and 30 minutes making a restaurant decision.

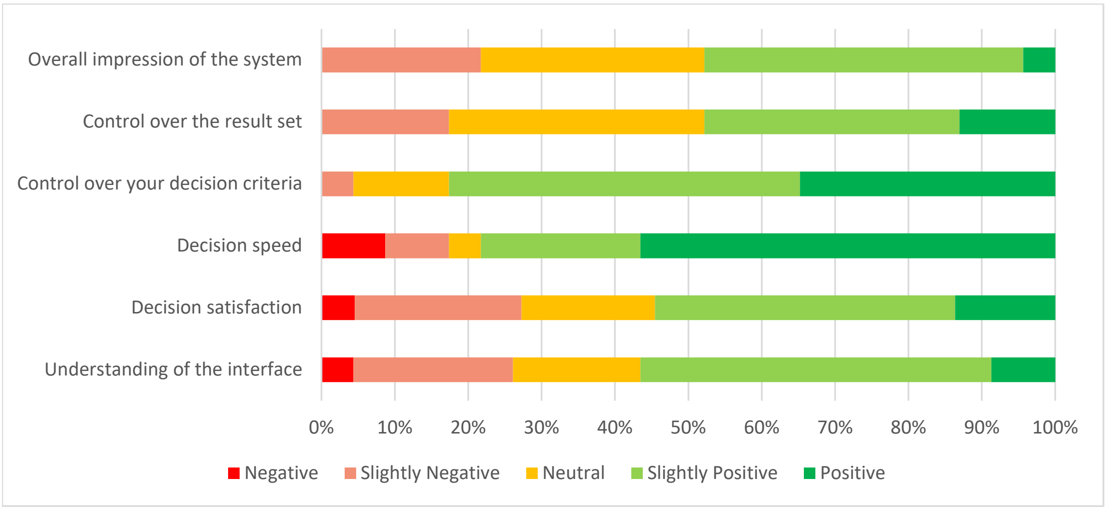


Figure 6.1: Survey responses: overall impressions of interface

6.3 Results

In the following section, responses from the user study survey will be presented. This will include data and impressions about the overall prototype, about the visualization interfaces and how they compare, as well as some general attitudes toward the application and concept. This section only includes the raw information, which is then followed by a discussion of the study and responses.

6.3.1 Overall Prototype

The first section of the survey asked respondents to evaluate the interface as a whole. These general questions were asked first to avoid any bias resulting from questions presented throughout the survey. This section focused primarily on general impressions of the interface and the concept of user control. Visualization-specific sections were included after the general section was complete. The responses for this section can be seen in Figure 6.1.

Overall, the impression users had on the system was relatively positive. Almost half (11 of 23 participants) had a slightly positive or positive overall impression of the system, whereas only 21.7% (5 of 23 users) had a slightly negative impression. No users rated the overall system as completely negative.

The results show that the interface was especially effective at producing fast decisions. Over half of the respondents (13 out of 23 users) had a positive impression of the system regarding decision speed, and 78.3% (18 of 23 users) had at least a slightly positive impression.

Additionally, users expressed a positive response with regards to control over their decision criteria. 82.6% of participants had either a slightly positive or positive impression of this aspect. Contrary to the decision speed attribute, no users expressed a completely negative impression of this, and only one participant rated it as slightly negative. However, despite the positive impression of control over criteria, the response

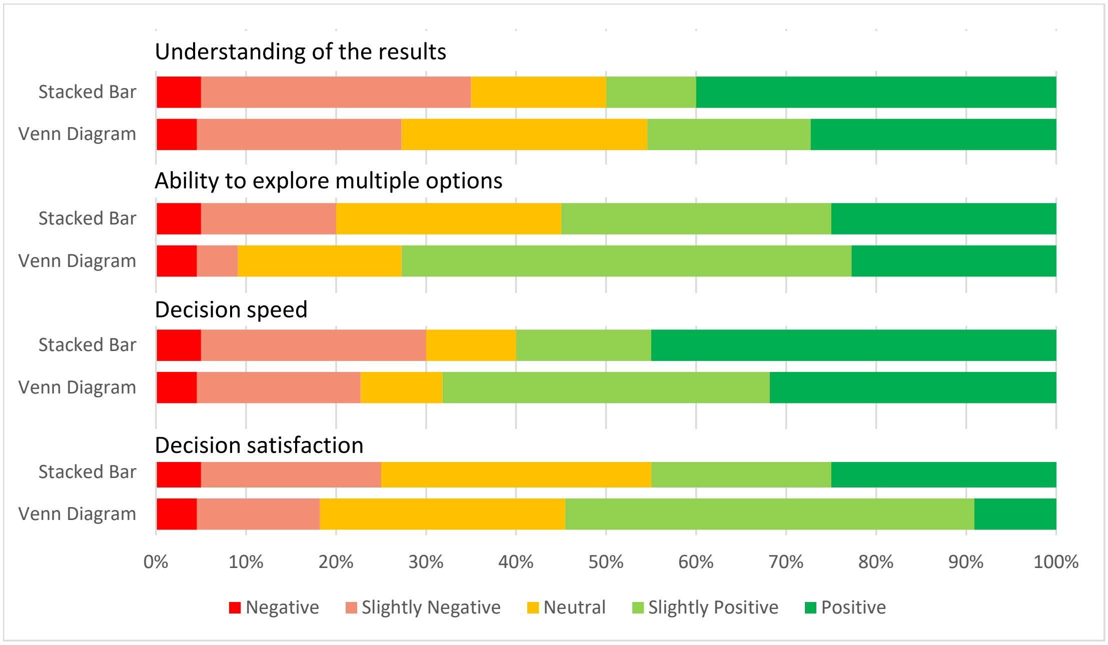


Figure 6.2: Survey responses: impressions of each visualization type

was less positive regarding control over the results themselves. While only 17.4% (4 of 23 responses) mentioned a slightly negative impression of control over the result set, it also drew the largest neutral response, with 34.8% (8 of 23 participants).

Decision satisfaction and understanding of the interface both drew similar attitudes. They were both slightly more positive than the overall impression of the system and control over the result set, with a slightly positive or positive rating by 54.5% and 56.5% of users respectively. Both categories also drew one entirely negative response.

6.3.2 Visualization Interfaces

To better understand how users viewed the visualizations, participants were asked to rate each interface type using a set of questions utilizing a Likert Scale from 1 to 5. The same categorizations were used independent of the type of visualization, and users were asked to respond relative to both interfaces, regardless of which one they evaluated first. The two sections were however presented in the same order as the evaluation based on an earlier response in the survey about which visualization type the respondent used first. The responses for each visualization type can be seen in Figure 6.2.

When comparing the two types of visualizations, the Stacked Bar interface caused a more polarizing opinion than the Venn Diagram. In all four responses, users had both a more positive and more negative opinion of the Stacked Bar interface. The Venn Diagram did receive higher levels of slightly positive responses in all categories except for understanding of the results. Both visualization types received generally positive responses in all categories. The only two items which received less than half of responses as slightly positive or positive were understanding of the results for the Venn Diagram, and decision satisfaction for the Stacked Bar.

Of the four categories, understanding of the results produced the most negative response for both visualization types. 35% (7 of 20 users) had a negative or slightly negative impression of understanding using the Stacked Bar, compared to 27.2% (6 of 22 users) for the Venn Diagram. The Stacked Bar Chart drew more overall positive impression as well as generally positive feedback in this area than the Venn Diagram. Half of the responses (10 of 20 participants) rated their understanding of the results as positive or slightly positive for the Stacked Bar, compared to 45.5% for the Venn Diagram.

Users rated the Stacked Bar Chart with more entirely positive responses, but fewer overall positive impressions when compared to the Venn Diagram for all three other categories: ability to explore multiple options, decision speed, and decision satisfaction. The largest disparity was in decision speed, where 45% (9 of 20 users) had an entirely positive impression for the Stacked Bar chart, compared to 31.8% (7 of 22 users) for the Venn Diagram. However, when combining positive and slightly positive results, the Venn Diagram received 68.2%, compared to only 60% for the Stacked Bar interface.

Over the study, participants were relatively split on their preference between the two visualization types, slightly leaning toward the Stacked Bar Chart. 52.2% of participants said they preferred the Stacked Bar Chart, with the remaining 47.8% choosing the Venn Diagram. Users tended to slightly prefer whichever visualization type they were first presented with. Participants exposed to the Stacked Bar Chart first preferred it 63.6% of the time, and those beginning with the Venn Diagram preferred it 58.3% of the time.

Looking from a more implicit angle, Figure 6.3 shows the overall impressions separated by which visualization type the participant first used. Interestingly, users who began the study with the Venn Diagram rated the system more positively in every category. Respondents starting with the Venn Diagram did not provide a single negative rating for either category regarding control. Over 90% of those users rated their control over decision criteria as positive or slightly positive. The only exceptions to this pattern were the overall impression of the system and control over decision criteria. In these two categories, users who started with the Stacked Bar chart were more likely to rate them as completely positive. However, both items still received a greater amount of slightly positive and positive responses from users starting with the Venn Diagram interface.

6.3.3 General Attitudes

In the final section of the study survey, participants were asked to evaluate the interface through various open-ended questions. This allowed respondents to provide valuable feedback that related more specifically to their personal experience.

One question asked participants if the prototype reminded them of any other existing systems. This question was asked first to avoid any recency bias. Only five respondents mentioned a previous system, the most common response being Yelp[2] (3 out of 23 participants). However, when later asked whether users had relied on various other restaurant decision-support systems, almost all participants said that they had used at least one. The most commonly used system was Google Maps, with 90% (18 of 23 participants) usage.

Users were asked what advantages this system provided compared to existing deci-

[2]https://www.yelp.com/

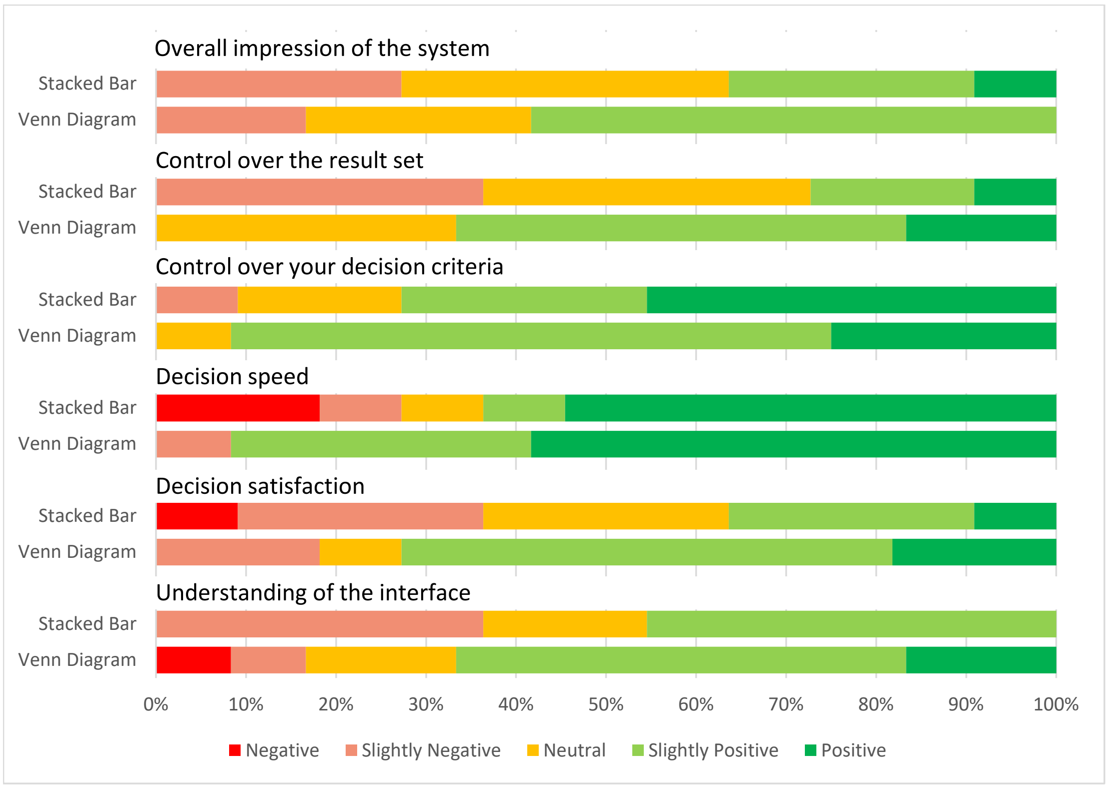


Figure 6.3: Survey responses: overall impressions filtered by visualization type first encountered by the participant

sion support tools on the web. Of the 22 respondents, 6 users (27.3%) mentioned the ability to attach weight to their criteria, and 4 others (18.2%) made a reference to the criteria selection in general. This means that nearly half of the users identified features relating to their control over the search terms. Additionally, 6 users also brought up the benefit of seeing the results in a visual display.

When asked what they liked best about the application, the most prevalent themes were about the speed, as well as ease of use and understanding. Over half the respondents (12 of 20 responses) mentioned something about one of these topics. Nearly half (45%) of responses to this question dealt with the idea of being able to control their decision criteria. Only a few users mentioned the aspect of weighting specifically. The visual results feature was mentioned by 30% (6 of 20 participants), with the Bar Chart and Venn Diagram each specifically receiving 1 mention.

Contrarily, users were also asked to identify disadvantages that this system would have compared to existing tools. Many of the responses tended toward lacking features such as ratings, additional restaurant information, or other interface-based deficiencies. A few responses indicated that the interface could have some learning curve or be difficult and confusing to understand on first attempt. However, as one user suggested, this "could be overcome by, time/use, or adding directions or a how to understand...section." A few participants mentioned that having additional guidance for search terms would be helpful. However, four users also responded that what they liked best about the application was the freedom to select their own criteria. For example, one user mentioned "the ability to search anything that is important to you rather than what is important to others" as their favorite feature.

When prompted for what they liked least about the interface, the visual results interface was mentioned heavily. 8 of the 22 responses mentioned something about the results display, with the Bar Chart and Venn Diagram specifically called out three times each. 13.6% (3 of the 22 responses) specifically mentioned issues with selecting which search criteria to use. Additionally, a few users seemed to miss some features for additional exploration of specific results which are common in other systems such as maps, restaurant links, or menus.

The final question of the survey asked users how the system could be improved. The most common theme in these responses was dealing with additional ability to explore the results. Over half of the answers (11 of 20 responses) mentioned improving the system by including things such as ratings, location information, menus, and other restaurant information. Related to searches, 15% (3 of 20 responses) mentioned including more well-defined attribute selection as a possible improvement.

6.4 Discussion

The previous sections in this chapter have introduced the structure, design, implementation, and results from the user study. The following sections will discuss what those results mean, and what factors may have influenced the data.

6.4.1 Issues and Limitations

As in many user studies, as well as prototype testing, a few issues arose during the evaluation period. During the study, a few technical and design issues appeared, which may have affected the results. However, there were no major functional issues to invalidate the results.

The most prevalent issue noticed was that a few issues with browser and platform compatibility, especially with mobile users, came into play. In the future, it would likely be best to limit users to specific browsers and platforms to those which have already been validated.

Another minor issue that came up during the study was an error in implementation which caused search terms leading with numbers to break the system. After the issue was brought up by a participant, it was immediately corrected to enhance the experience for users. As a result of this mid-study change, it is possible that some participants did not receive the exact same experience.

After four responses in the survey, it was apparent that the survey flow was incorrect. Users who began the study with the Venn Diagram visualization were not presented with questions about the Stacked Bar Chart. Therefore, three of the responses did not contain answers regarding that interface. However, it is unlikely that this would cause any issues. To prevent contamination from these results, any data comparisons dependent on the presentation order of visualizations, these responses were not included in the analysis throughout this chapter.

With regard to the study participants, the sample size of 23 was large enough to provide tendencies of a potential user base, but likely is still too small to produce a high level of confidence in the results. Additionally, users participated strictly on a volunteer basis and were all procured through personal connections with the author of this study. Though this is believed to be a generally representative sample of the potential user base for a system such as this, a more randomly selected group of participants would provide more stable and reliable results.

6.4.2 Result Analysis

Despite these issues and limitations, it is possible to draw some valuable conclusions from the results set. Based on the user study performed, it appears that participants had a generally positive attitude toward the system. Additionally, there was an especially good impression of the control offered by the system over the results. Though the response was not as enthusiastic toward the visualization techniques of result display, there was also little negative reaction.

From a comparison point of view, the Stacked Bar Chart was generally more well-received when looking at explicit user responses. However, the participants tended to respond more positively to the system as a whole when presented with the Venn Diagram interface first. One explanation for this could be that the execution of the Venn Diagram interface was not as well-received, but users did like the concept of the display. Another possible reason for this disparity is that users got more used to the system through use and ended up preferring the second system used as a result. However, this is counteracted by results showing that participants identified their preferred interface as the one which was presented to them first, regardless of which that was.

One of the primary goals of the study was to identify how users felt control over the system impacted their decisions. Interestingly, while there did not seem to be a connection between control over their decision criteria and impressions of the system, there was a noticeable positive correlation between control over the result set when compared with the overall impression of the system, decision speed, and decision satisfaction. Figure 6.4 shows the average of these ratings on the vertical axis with the ratings for control over the result set on the horizontal axis.

When looking at the open-ended qualitative responses, many of the responses to both the visual results and user control were positive. The survey results show that people were especially satisfied with their ability to control their searches and the possibility to add weights to different attributes. Additionally, many of the negative interpretations of the system tended to point out deficiencies in functionality when compared to other existing decision system. For example, multiple responses suggested that adding additional restaurant information such as menus or locations would be useful. This supports the likelihood that utilizing similar control and visualization-based applications alongside with other existing functionality would likely lead to a higher level of satisfaction.

Another common trend in the results of the survey were around decision speed and ease of use. With respect to general impressions of decision speed, most responses were positive. Additionally, users tended to mention the simplicity and efficiency of the system when asked about positive aspects. As a result, it can be argued that using visual techniques as well as improving control over the system could be valuable in creating more efficient and easy-to-use decision support systems in the future. However, care

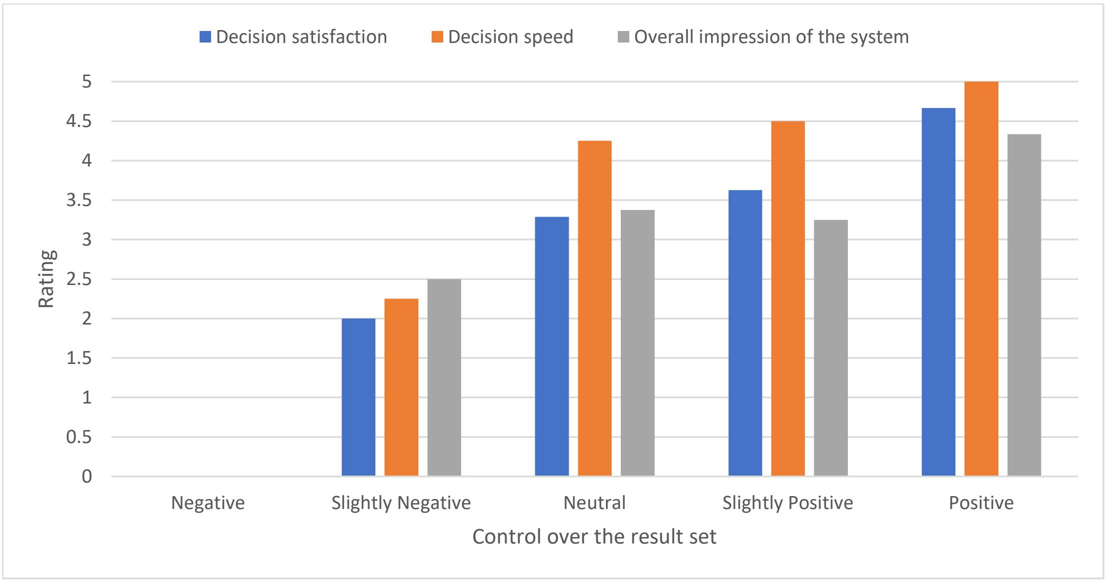


Figure 6.4: Survey responses: overall impression, decision speed, and decision satisfaction ratings based on rating of control over the result set

must also be taken in that this system was only a prototype focused on these specific attributes. It is possible that the positive response to these features was largely due to a lack of other options that might be available in a more complex application.

Though satisfaction and speed were generally high, there were some users who expressed concern about understanding of the results. A higher number of participants had a positive association with control over their own criteria when compared to control over the result set. As the earlier chart shows, being able to increase the sense of control over the results themselves would likely lead to a higher level of satisfaction. Additionally, providing more explanation of how the algorithm works and improved instruction about the system could improve the application. A few users mentioned confusion and learning curves, showing that additional guidance and transparency may increase the satisfaction of users.

Chapter 7

Conclusion

As decision making becomes more sophisticated through various Web tools and systems, this paper investigates how applications using visual techniques and improved user control could influence the decision process. Through previous studies, it has been shown that many factors change how users view decisions on the Web. During the decision process, people tend to consider different attributes and criteria before making a decision. Additionally, how systems are evaluated can depend on different situations and use cases, as well as differences between individuals. Though systems seem to be moving more toward artificial intelligence and machine learning to provide users with more effective recommendations and search results, these situational and personal factors suggest that providing a more custom and controllable approach might be feasible. Prior studies have shown that increased user control often leads to positive attitudes toward decision satisfaction and trustworthiness.

To further explore how interaction, control, and visual results can affect user attitudes toward Web-based decision making, a prototype was created which focused primarily on these aspects. In this prototype, users have the ability to search for multiple open-ended queries to find restaurant results in a specific location. Once a second search attribute is added, the possibility to control how much each one influences the result set is available through a slider interface. This provides a unique, user-centric experience which offers additional control over decision criteria and the results themselves. Once the searches are entered, results are displayed in one of two visualizations, allowing users to quickly see and interpret their possible options. These visual result displays are updated instantly and automatically any time a query is added or removed, or if the attribute weight is modified.

Two visual interfaces are utilized in the application to provide alternate exploration capabilities, allowing a comparison between different types of visualizations, as well as a better insight into the general attitude toward visual results. One visualization presents a Stacked Bar Chart, where each bar represents one of the top ten restaurant results. Each restaurant's bar contains separately colored segments, each representing one of the search attributes. The second interface focuses more on the interaction between the different queries, with top restaurants each represented in a Venn Diagram. The overall combined score is represented through node opacity, and nodes fall within the different regions of the diagram depending on which attributes match.

To evaluate the prototype, as well as general attitudes toward the core concepts of user control and visual display techniques, a user study was conducted. Participants were asked to complete three scenarios using both visual interfaces and then take a survey based on their experiences. The study revealed that respondents generally had positive attitudes toward the interface. Users rated control over their decision criteria, as well as decision speed, especially positively. A slight majority identified the Stacked Bar Chart as their preferred interface, though participants who began the study using the Venn Diagram tended to have more positive attitudes toward the system in general.

In addition to rating various attributes of the system on a Likert Scale from 1 to 5, participants were asked qualitative open-ended questions about the concept. When asked to identify advantages of the system, many users mentioned the search queries, specifically relating to the ability to control their weighted influence on the results. The most common negative aspects pointed out tended to lean in the direction of additional details, especially those in similar existing systems. Multiple respondents mentioned including additional restaurant information such as menus or ratings. Additionally, some participants mentioned experiencing some confusion and learning curve when first beginning the study. This indicates that perhaps additional explanation of the interface and results would benefit the application, but that the core concepts of user control and visualization generally elicited a positive reaction.

7.1 Future Work

This prototype and investigation provides some base-level evaluation of user attitudes toward control and visualization within the decision-making process on the Web. However, it has also invoked a number of potential future explorations.

One of the more common themes in the user study responses was in the lack of information. As this prototype sought to focus primarily on the core concepts, it also lacked many familiar features that users may be used to from similar applications. It would be interesting to investigate how adding these additional features might influence attitudes toward user control and visualization. The positive reaction to these concepts in this study suggest that combining them with existing possibilities would provide additional satisfaction. However, adding these features would also add complexity to the system, and may distract from some of the positive attributes of this prototype.

A major pattern within responses during the survey was the idea of query generation. Some users mentioned that they liked having the freedom to enter any search term, which corresponded more directly to their needs as opposed to what the system thinks they want. However, others also mentioned that having more guidance as to what sort of attributes can be added could be helpful. How could we more deeply understand how users can produce the most relevant and useful queries? Ultimately, numerous studies have been done on providing the best results based on various input data, but there has been much less exploration into how systems can help users produce the best input. One example of a potential option is combining existing faceted systems with the concept of weighted attributes. For example, hotel search sites such as Booking.com[1] contain various ratings for attributes such as location, safety, Wifi connection, and more. Allowing users to control which of these items is more important to them is

[1]https://www.booking.com/

however usually not offered. Perhaps providing users with certain facets such as prices or categories can provide more accurate results, but maybe giving the user more freedom in searching is beneficial to their overall satisfaction.

Another future expansion possibility is in the realm of visualizations. The two visualization types in this paper were relatively simple displays. Additionally, the survey results showed that participants did not generally feel particularly more drawn toward one type. Responses indicated that the general attitudes toward visual results were positive, but there was also a moderate amount of confusion upon first testing, as well as complaints of missing information. Finding visual techniques that provide a more intuitive interface, as well as the possibility to do more exploration and interaction with the system might provide users with a more satisfactory experience.

During the design of the system, one of the primary objectives was to maintain the potential for extension. One planned future exploration is the integration of different options with regard to the actual result inputs. The system was built and evaluated based on a specific type of input (searching) in a specific domain (restaurants). During the development, an important consideration was creating a modular and flexible environment that could be expanded upon in the future. Search results through a public API were used in this prototype, but perhaps importing results from other media would be an interesting exercise. Some potential other input sources could be ranked listings from experts, objective statistical ratings, or other Recommender Systems.

One major challenge in integration between various data sources is in matching and aggregating the various lists. Schema matching is a field that has been investigated many times. Finding a straightforward method of integrating these sources through schema matching would be an important step in advancing the prototype developed into a more powerful position as a decision support system.

Finally, also included in the adapter component development of this prototype was the possibility to include a custom scoring function. This particular application was dependent on rankings from ordered lists for calculating the overall score for items. However, integrating a more complex algorithm which can include potential external rating systems or other inputs would improve the usefulness of the system.

References

Literature

[1] B. Bazeer Ahamed and T. Ramkumar. “An intelligent web search framework for performing efficient retrieval of data”. *Computers & Electrical Engineering* 56 (2016), pp. 289–299 (cit. on p. 10).

[2] Svetlin Bostandjiev, John O’Donovan, and Tobias Höllerer. “TasteWeights: A Visual Interactive Hybrid Recommender System”. In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. RecSys ’12. Dublin, Ireland: ACM, 2012, pp. 35–42 (cit. on pp. 13, 16).

[3] Yu-Chen Chen, Rong-An Shang, and Chen-Yu Kao. “The effects of information overload on consumers’ subjective state towards buying decision in the internet shopping environment”. *Electronic Commerce Research and Applications* 8.1 (2009), pp. 48–58 (cit. on pp. 4, 8, 10).

[4] Deng-Neng Chen et al. “A Web-based personalized recommendation system for mobile phone selection: Design, implementation, and evaluation”. *Expert Systems with Applications* 37.12 (2010), pp. 8201–8210 (cit. on p. 11).

[5] Stavin Deeswe and Raymond Kosala. “An Integrated Search Interface with 3D Visualization”. *Procedia Computer Science* 59 (2015). International Conference on Computer Science and Computational Intelligence (ICCSCI 2015), pp. 483–492 (cit. on p. 12).

[6] Michael D. Ekstrand et al. “Letting Users Choose Recommender Algorithms: An Experimental Study”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. Vienna, Austria: ACM, 2015, pp. 11–18 (cit. on p. 16).

[7] Fatih Gedikli, Mouzhi Ge, and Dietmar Jannach. “Understanding Recommendations by Reading the Clouds”. In: *E-Commerce and Web Technologies: 12th International Conference, EC-Web 2011, Toulouse, France, August 30 - September 1, 2011. Proceedings*. Ed. by Christian Huemer and Thomas Setzer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 196–208 (cit. on p. 13).

[8] Fatih Gedikli, Dietmar Jannach, and Mouzhi Ge. “How should I explain? A comparison of different explanation types for recommender systems”. *International Journal of Human-Computer Studies* 72.4 (2014), pp. 367–382 (cit. on pp. 7, 11).

[9] E. Gomez-Nieto et al. “Similarity Preserving Snippet-Based Visualization of Web Search Results”. *IEEE Transactions on Visualization and Computer Graphics* 20.3 (Mar. 2014), pp. 457–470 (cit. on p. 12).

[10] Poonam Goyal and N. Mehala. “Concept Based Query Recommendation”. In: *Proceedings of the Ninth Australasian Data Mining Conference - Volume 121*. AusDM ’11. Ballarat, Australia: Australian Computer Society, Inc., 2011, pp. 69–78 (cit. on p. 8).

[11] Chen He, Denis Parra, and Katrien Verbert. “Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities”. *Expert Systems with Applications* 56 (2016), pp. 9–27 (cit. on pp. 12, 13, 15).

[12] Vagelis Hristidis and Yannis Papakonstantinou. “Algorithms and applications for answering ranked queries using ranked views”. *The VLDB Journal* 13.1 (Jan. 2004), pp. 49–70 (cit. on p. 9).

[13] Choon Hui Teo et al. “Adaptive, Personalized Diversity for Visual Discovery”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys ’16. Boston, Massachusetts, USA: ACM, Sept. 2016, pp. 35–38 (cit. on p. 11).

[14] Ihab F. Ilyas, George Beskales, and Mohamed A. Soliman. “A Survey of Top-k Query Processing Techniques in Relational Database Systems”. *ACM Computing Surveys* 40.4 (Oct. 2008), 11:1–11:58 (cit. on p. 8).

[15] Yuanchun Jiang, Jennifer Shang, and Yezheng Liu. “Maximizing customer satisfaction through an online recommendation system: A novel associative classification model”. *Decision Support Systems* 48.3 (2010). New concepts, methodologies and algorithms for business education and research in the 21st century, pp. 470–479 (cit. on p. 11).

[16] Thorsten Joachims. “Optimizing Search Engines Using Clickthrough Data”. In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’02. Edmonton, Alberta, Canada: ACM, 2002, pp. 133–142 (cit. on p. 8).

[17] Antti Kangasrääsiö, Dorota Glowacka, and Samuel Kaski. “Improving Controllability and Predictability of Interactive Recommendation Interfaces for Exploratory Search”. In: *Proceedings of the 20th International Conference on Intelligent User Interfaces*. IUI ’15. Atlanta, Georgia, USA: ACM, 2015, pp. 247–251 (cit. on p. 15).

[18] Bart P. Knijnenburg, Niels J.M. Reijmer, and Martijn C. Willemsen. “Each to His Own: How Different Users Call for Different Interaction Methods in Recommender Systems”. In: *Proceedings of the Fifth ACM Conference on Recommender Systems*. RecSys ’11. Chicago, Illinois, USA: ACM, 2011, pp. 141–148 (cit. on pp. 10, 15).

[19] Joseph A. Konstan and John Riedl. “Recommender systems: from algorithms to user experience”. *User Modeling and User-Adapted Interaction* 22.1 (Apr. 2012), pp. 101–123 (cit. on p. 15).

[20] Lin Li et al. “Random walk based rank aggregation to improving web search”. *Knowledge-Based Systems* 24.7 (2011), pp. 943–951 (cit. on p. 9).

[21] Yung-Ming Li, Chun-Te Wu, and Cheng-Yang Lai. “A social recommender mechanism for e-commerce: Combining similarity, trust, and relationship”. *Decision Support Systems* 55.3 (2013), pp. 740–752 (cit. on p. 10).

[22] Rensis Likert. “A Technique for the Measurements of Attitudes”. *Archives of Psychology* 22.140 (1932), pp. 1–55 (cit. on p. 40).

[23] Jie Lu et al. “Recommender system application developments: A survey”. *Decision Support Systems* 74 (2015), pp. 12–32 (cit. on pp. 6, 9).

[24] Matteo Palmonari et al. “Aggregated search of data and services”. *Information Systems* 36.2 (2011). Special Issue: Semantic Integration of Data, Multimedia, and Services, pp. 134–150 (cit. on p. 9).

[25] Denis Parra, Peter Brusilovsky, and Christoph Trattner. “See What You Want to See: Visual User-driven Approach for Hybrid Recommendation”. In: *Proceedings of the 19th International Conference on Intelligent User Interfaces.* IUI ’14. Haifa, Israel: ACM, 2014, pp. 235–240 (cit. on pp. 13, 16).

[26] Jeffrey Pound, Stelios Paparizos, and Panayiotis Tsaparas. “Facet Discovery for Structured Web Search: A Query-log Mining Approach”. In: *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data.* SIGMOD ’11. Athens, Greece: ACM, 2011, pp. 169–180 (cit. on p. 14).

[27] Erhard Rahm and Philip A. Bernstein. “A survey of approaches to automatic schema matching”. *The VLDB Journal* 10.4 (Dec. 2001), pp. 334–350 (cit. on p. 9).

[28] Francesco Ricci et al. *Recommender Systems Handbook.* New York, NY, USA: Springer-Verlag New York, Inc., 2010 (cit. on p. 11).

[29] Michael Scholz et al. “A configuration-based recommender system for supporting e-commerce decisions”. *European Journal of Operational Research* 259.1 (2017), pp. 205–215 (cit. on p. 10).

[30] Barry Schwartz. *The Paradox of Choice: Why More is Less.* New York, NY, USA: Harper Collins, 2004 (cit. on p. 10).

[31] Madeleine Sorapure. “Information Visualization, Web 2.0, and the Teaching of Writing”. *Computers and Composition* 27.1 (2010). Composition 2.0, pp. 59–70 (cit. on p. 12).

[32] Nava Tintarev and Judith Masthoff. “Evaluating the effectiveness of explanations for recommender systems”. *User Modeling and User-Adapted Interaction* 22.4 (Oct. 2012), pp. 399–439 (cit. on pp. 9, 11).

[33] Damir Vandic, Jan-Willem van Dam, and Flavius Frasincar. “Faceted product search powered by the Semantic Web”. *Decision Support Systems* 53.3 (2012), pp. 425–437 (cit. on p. 10).

[34] Katrien Verbert et al. “Visualizing Recommendations to Support Exploration, Transparency and Controllability”. In: *Proceedings of the 2013 International Conference on Intelligent User Interfaces.* IUI ’13. Santa Monica, California, USA: ACM, 2013, pp. 351–362 (cit. on pp. 13, 16).

[35] Martin Voigt et al. “Weighted Faceted Browsing for Characteristics-based Visualization Selection Through End Users”. In: *Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems*. EICS ’12. Copenhagen, Denmark: ACM, 2012, pp. 151–156 (cit. on p. 14).

[36] Ling-Ling Wu, Ya-Lan Chuang, and Yuh-Jzer Joung. “Contextual multi-dimensional browsing”. *Computers in Human Behavior* 24.6 (2008). Including the Special Issue: Electronic Games and Personalized eLearning Processes, pp. 2873–2888 (cit. on p. 14).

Online sources

[37] Jakob Nielsen. *Converting Search into Navigation*. Mar. 2013. URL: https://www.nngroup.com/articles/search-navigation/ (cit. on p. 8).

Check Final Print Size

— Check final print size! —

width = 100mm
height = 50mm

— Remove this page after printing! —