"Every Website Is a Puzzle!": Facilitating Access to Common Website Features for People with Visual Impairments

NATÃ M. BARBOSA, University of Illinois at Urbana–Champaign
JORDAN HAYES, Syracuse University
SMIRITY KAUSHIK and YANG WANG, University of Illinois at Urbana–Champaign

Navigating unfamiliar websites is challenging for users with visual impairments. Although many websites offer visual cues to facilitate access to pages/features most websites are expected to have (e.g., log in at the top right), such visual shortcuts are not accessible to users with visual impairments. Moreover, although such pages serve the same functionality across websites (e.g., to log in, to sign up), the location, wording, and navigation path of links to these pages vary from one website to another. Such inconsistencies are challenging for users with visual impairments, especially for users of screen readers, who often need to linearly listen to content of pages to figure out how to access certain website features. To study how to improve access to main website features, we iteratively designed and tested a command-based approach for main features of websites via a browser extension powered by machine learning and human input. The browser extension gives users a way to access high-level website features (e.g., log in, find stores, contact) via keyboard commands. We tested the browser extension in a lab setting with 15 Internet users, including 9 users with visual impairments and 6 without. Our study showed that commands for main website features can greatly improve the experience of users with visual impairments. People without visual impairments also found command-based access helpful when visiting unfamiliar, cluttered, or infrequently visited websites, suggesting that this approach can support users with visual impairments while also benefiting other user groups (i.e., universal design). Our study reveals concerns about the handling of unsupported commands and the availability and trustworthiness of human input. We discuss how websites, browsers, and assistive technologies could incorporate a command-based paradigm to enhance web accessibility and provide more consistency on the web to benefit users with varied abilities when navigating unfamiliar or complex websites.

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

Additional Key Words and Phrases: Website accessibility, intelligent personal assistants, website commands

ACM Reference format:
Natã M. Barbosa, Jordan Hayes, Smirity Kaushik, and Yang Wang. 2022. *"Every Website Is a Puzzle!"*: Facilitating Access to Common Website Features for People with Visual Impairments. *ACM Trans. Access. Comput.* 15, 3, Article 19 (July 2022), 35 pages.
https://doi.org/10.1145/3519032

This work was supported in part by the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant 90DP0061-01-00) and the National Science Foundation (NSF grant CNS-1652497).
Authors' addresses: N. M. Barbosa, S. Kaushik, and Y. Wang, University of Illinois at Urbana–Champaign, 501 E Daniel St, Champaign, IL 61820; emails: {natamb2, smirity2, yvw}@illinois.edu; J. Hayes, Syracuse University, Syracuse, 343 Hinds Hall, Syracuse, NY 13244; email: jhayes05@syr.edu.

1936-7228/2022/07-ART19 $15.00
https://doi.org/10.1145/3519032

1 INTRODUCTION

Navigating websites requires disparate effort from users with visual impairments [15, 23, 69]. When browsing complex websites with a screen reader (e.g., online shopping), users with visual impairments can get overwhelmed and confused due to the way in which websites are rendered, often having to deal with inaccessible content and parsing long lists of links. When navigating an unfamiliar website, common features found on many websites conveyed via visual cues are not immediately accessible to users with visual impairments. For example, when visiting a website for the first time, one may expect to find the login and sign-up links somewhere near the top-right corner of a website, and contact links appearing toward the bottom. However, when using screen readers, users may not be able to rely on such visual conventions, as (1) the content is narrated on a linear fashion based on the document-level structure of the page and (2) there is no enforced standard about how the desired feature might be implemented (e.g., is the link "log in," "sign in," or "my account?"), requiring users to engage in trial-and-error strategies [69]. Such lack of consistency causes frustration and inefficiency among users with visual impairments, preventing them from experiencing the web in ways others might do with less effort. For instance, Figure 1 shows the diversity of the wording of hyperlinks of two features, log in and store finding, across some popular websites in the United States. These discrepancies can be challenging for users when first visiting websites: they need to learn (and later recall) each website's wording, which can diminish users' confidence and independence [61]. Therefore, more consistency could benefit users with visual impairments.

Recent studies involving **intelligent personal assistants (IPAs)** (e.g., [1, 2, 46, 61]) hint at an emerging paradigm of giving users with visual impairments ways to "actively solicit" features and information they desire as an alternative to the long-established "passive listener" [61] approach to accessibility. As an illustration, in the current paradigm, websites are parsed by users as a directory of links: screen reader users rely heavily on keyboard shortcuts to navigate links and page sections. This passive listening paradigm has been shown to overwhelm users of assistive technologies [69]. Alternatively, *active solicitation* [61] allows users to focus on their goals and provides consistency across websites, enabling users to easily access the resources they need to complete the task at hand without having to deal with individual design differences. Arguably, users of screen readers have to "figure out" how each website is structured to access a certain common feature (e.g., log in, contact, deals). In contrast, enabling users to actively solicit higher-level features could provide a consistent way for them to browse, in which more focus is given to the task at hand rather than requiring users to navigate the content and structure of each website effectively "figure out" how to access common features on each website's unique design.

To address this issue, we aimed to study whether command-based browsing for main website features—especially common features across websites—could be a beneficial interaction modality for users with visual impairments. By using commands to access major features of websites (e.g., sign up, contact, find nearby stores), users would not have to deal with website design variations or learn the structure of each individual website. To test whether this paradigm would benefit users with visual impairments, we iteratively designed and evaluated a system allowing users to actively solicit high-level features they wish to access on the present website, which are mapped to target pages via machine learning models and human input. We designed, implemented, and evaluated our system in a lab setting, starting with a Wizard-of-Oz prototype, then building and evaluating a functional system. Inspired by universal design—the idea of designs for underserved groups benefiting everyone [43, 62, 63]—we also tested our system with users without visual impairments.

We found that our system enabled users to access common features across websites more quickly (measured by logs) and directly (based on qualitative feedback). Participants thought the command-based approach afforded added consistency and reduced the number of steps and unexpected

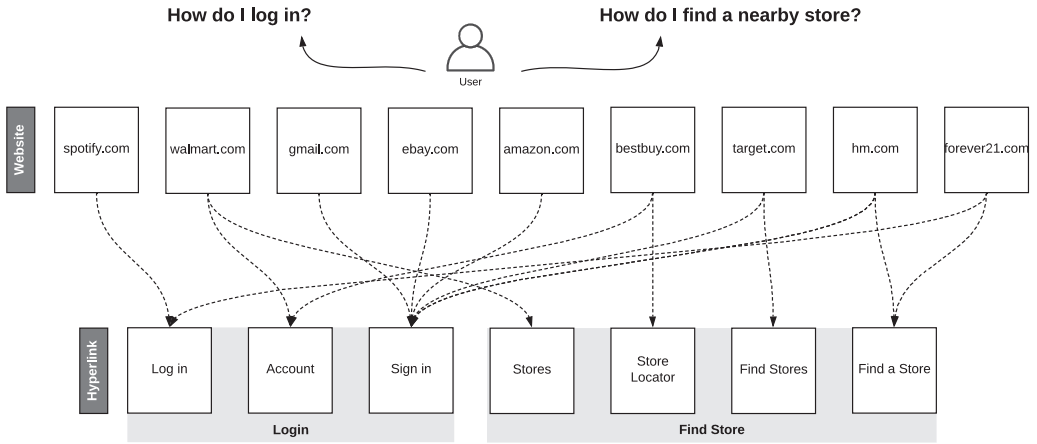

Fig. 1. Diversity of hyperlink wording for common features across popular websites in the United States. Although visual shortcuts are present (e.g., log in at the top right or left, store finding near the top), these are not accessible to users with visual impairments, thus requiring users to learn each website. Our goal is to support access to common features via commands in a consistent way, even if they are implemented differently.

challenges faced along the way. The system was found to be most useful on unfamiliar, cluttered, or infrequently visited websites or features, even by users without visual impairments. Participants also wished to use "crowd-favorite" website features and human input from affinity groups while attempting to access a main website feature. Participants raised some concerns about website discovery, privacy and security, and the handling of unsupported features and commands.

Drawing from our findings, we discuss how machine learning and human input can assist command-based browsing and present design implications for personal assistants and assistive technologies. We also discuss how implementing a "consistency" infrastructure and/or metadata for main, common website features could enable personal assistants to interact with individual websites on behalf of users in an active solicitation paradigm.

Our main contribution is showing that active solicitation of common website features is an effective interaction modality for users with visual impairments, providing consistency and preventing usability challenges faced due to the diversity of designs found across the web.

2 RELATED WORK

2.1 Making Web Browsing More Accessible

Navigating the web can be a daunting activity for users with visual impairments. Unexpected distractions, information overload, inaccessible content, and a lack of consistency make browsing the web quite challenging for this user group [15, 23, 69]—especially when completing tasks that may involve visiting multiple pages [13, 23, 28, 69]. As a result, users with visual impairments adopt strategies such as memorizing links [69] and exhaustively scanning a page [15, 69]. Such strategies can be very cognitively demanding, more so when inter-page navigation is required, such as when users must navigate several pages and websites to achieve their goal, and on unfamiliar websites [15, 61], as websites vary greatly in their structure and design.

Several approaches have been developed and evaluated to assist users with visual impairments in completing tasks online, including programming by demonstration to automate repetitive tasks (e.g., [11, 14, 40, 49]) (i.e., recording and reproducing a series of steps), using how-to knowledge repositories (e.g., [13, 40]), simplifying interactions with a given page's content and user interface

elements (e.g., [9, 44, 51, 66]), automating individual interactions with page elements such as "click search button," via voice commands [8, 9], and re-narrating or re-structuring the content of web pages in ways suitable to assistive technology users [16, 21, 47, 52].

Most notably, BrowseWithMe by Stangl et al. [61] is an intelligent online assistant created to help users with visual impairments shop for clothes on different websites. The system works by automatically parsing product web pages and converting content into structured representations that can be accessed by users via questions (i.e., commands) such as *"what is the price?"* or *"can I see a magnified image of the pants?"* across different shopping websites (e.g., a standard). Through a user study, the authors identified several advantages in the paradigm shift from *"treating users with visual impairments as passive listeners of unparsed information"* to *"giving users the ability to actively solicit desired information."* While specifically focused on shopping and on product pages, their interviews and user studies uncovered many benefits of command-based browser assistants more broadly, such as the consistency and effectiveness provided via commands that work across different websites, and increased trust and independence when interacting with features of websites.

Like in BrowseWithMe [61], a common, overlapping theme of these prior works is a departure from requiring users to linearly consume content, shifting toward more command-based and conversational approaches to web browsing for users with visual impairments. For example, the ability to interact via commands has been shown to be a promising modality for users with visual impairments, such as in accessing key sections of a given web page [28] and performing the next action on a given web page, such as when booking flights [7], and interacting with virtual world games [26]. This type of active solicitation modality has also shown promise in use of IPAs (e.g., Siri, Alexa) by users with visual impairments [1].

In our work, we study how to support quick and consistent (i.e., standardized) access to main, common website features (e.g., signing up, finding deals, finding popular items, and checking pricing information) across different websites. For example, users wanting to reset their password on websites that have user accounts would enter the "reset password" command and then be taken directly to the page where they could do so—regardless of website design. This approach is fundamentally different from the works presented earlier because it is not focused on page-level interactions (e.g., click button X) but website-level features (e.g., find nearby stores). For example, most prior works that share similar command-based or conversational approaches are scoped to interactions within a given web page and its user interface elements, whereas the approach we evaluated in this work aims at giving users quick access to the entry points for the most commonly expected features to be supported by websites. In facilitating access to main, common website features, it is our hope to (1) provide a layer of abstraction and consistency for users to access the main features of websites as they browse the web and (2) surface the most important or common features as users encounter new, unfamiliar websites or deal with website re-designs.

We note that some of the prior works could indirectly provide similar consistency. For example, TrailBlazer [13] could store scripts for each website's main features that still work on a different, albeit very similar, website where the script still works. In contrast, with our approach, such main features could be automatically recognized at any website visited without requiring manual annotations from website users, which is a scalability advantage. Similarly, in the work of Ashok et al. [8, 9], users could use the *Navigate* command to access portions of the present page, which could coincidentally relate to main website features (e.g., login forms), yet its scope is to facilitate granular, page-level interactions, not website-level access to features like in the approach we explore in this article.

Despite the aforementioned distinctions, we build our work from promising directions identified in prior works such as not completing tasks on behalf of the users (e.g., filling in forms) [48], allowing users to specify "commands" (i.e., their goals), leveraging natural language to allow users

to express high-level goals in a more conversational manner [50], and leveraging the wisdom of the crowd [28]. We note that in combination, these directions align well with the paradigm of active solicitation over sequentially narrating or describing content of websites, which we endorse. It is our hope that our work encourages a feasible path to more standardized access to website-wide features for people with visual impairments, complementing the numerous existing solutions for interacting with individual web pages.

2.2 Goal-Oriented Browsing

Not applicable only to users with visual impairments, prior work has shown that users browse websites for a variety of purposes, with a large amount of browsing activities consisting of transactions [36]. Thus, molding browsing to purpose can support many interactions online, especially when users make purposeful use of the web [42] (i.e., going online to complete certain tasks). Arguably, when supporting access to main features of websites such as "sign up," "check pricing," or "find store," executing such commands would directly support higher-level user goals. Nonetheless, no comprehensive user studies have been conducted to further evaluate this approach, resulting in a dearth of user evaluations on how goal-oriented browsing would work in practice and for whom it can be most beneficial. Most prior works that consider user intent when browsing the web are in the realm of query intent in search engines (e.g., [64]), but opportunities exist to support user intents from *within* the website—that is, how to support browsing goals once a user is already at a website, such as informational (e.g., "phone number"), navigational (e.g., "help page"), and transactional (e.g., "reset password"), within-website tasks.

To the best of our knowledge, the closest work to ours is a goal-oriented browser presented by Faaborg and Lieberman [25], which relied on programming by demonstration and a system for detecting high-level user goals on the *current* page based on semantic knowledge bases. They combined both approaches to map content on the current page to potential high-level user goals. For example, in seeing "cornstarch" on a recipe web page, the browser would add an "order food" hyperlink to the word, pointing to the user's favorite grocery store.

Our work is different from the system of Faaborg and Lieberman [25] in several ways. First, our system aims at achieving user goals *within* the scope of the website—that is, supporting access to features of the current website (e.g., find popular items) while also supporting access to features that may be common across most websites (e.g., sign up, contact). Second, in their system, the mapping between content and goal is based on semantic networks, whereas in our system, we use supervised learning models that do not require the maintenance of semantic constructs. Third, their system worked in a passive manner by identifying potential goals based on the content of the page, whereas our system allows users to actively solicit the desired features within the scope of the website. Fourth, they conducted a preliminary user evaluation with technically oriented users (the majority being programmers), whereas our evaluation includes users with diverse backgrounds and abilities. We also do not use programming by demonstration, which has limitations such as the inability to deviate from sequences [51] and lack of user control [48]. Last, but not least, we also use complementary human input when mapping features to target pages to overcome limitations of machine learning.

3 THE SYSTEM

Our system was designed to support command-based browsing for main, common features of websites. We define "intent" as a user's need to access a specific feature of a website. Throughout the article, we often use "intent" and "feature" interchangeably. As seen in Figure 1, the hyperlinks to the website features are implemented differently across websites, but visual conventions and cues can make access to these features a trivial task. For instance, the login link is likely to be

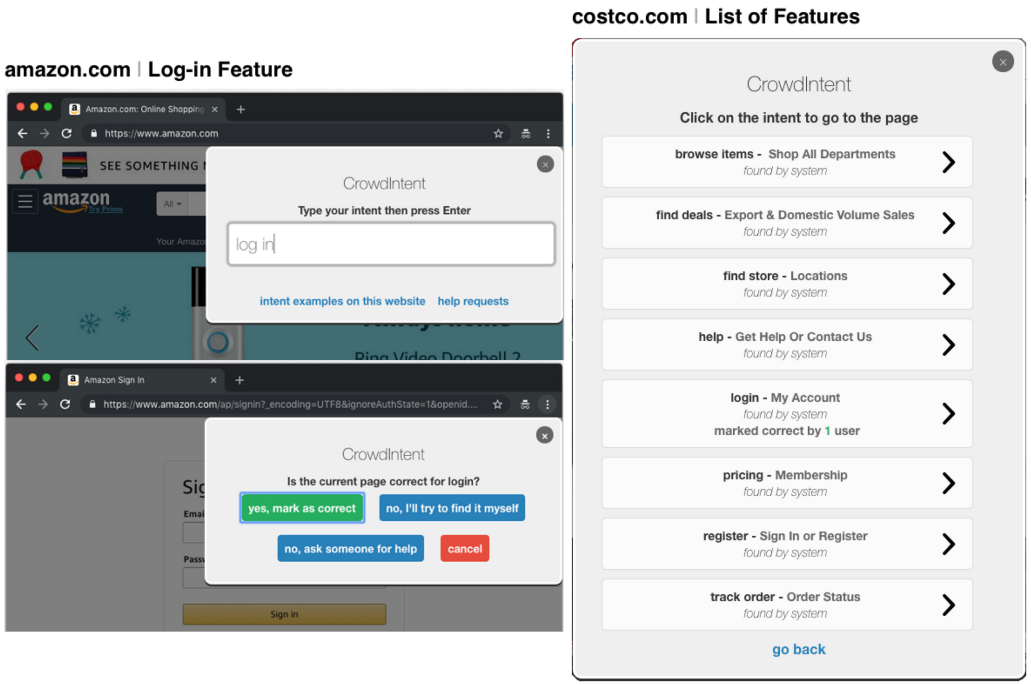

Fig. 2. Features can be accessed via direct input (left) or via the list of available features (right) as detected by the system (via machine learning) or prior annotations of users. On the left, users enter their intent—log in—on Amazon.com (i.e., the desired feature), and on the right, users choose from a list of available features on Costco.com that were detected by the system automatically. Upon entering or choosing the feature, users are taken to the target page, and mark it as correct, find the page manually, or get help from another user.

at the top-right corner of the website, and on retail websites, one can expect to find a link to find stores either near the top or the bottom. Unfortunately, a user browsing with a screen reader cannot leverage such cues and would have to explore and learn how the feature is denoted on each website by attempting to search for content on the page or browsing lists of hyperlinks via a trial-and-error approach (e.g., is it "log in" or "sign in?" Is it "store locations" or "find stores?"). Our system aims at overcoming these design variations via machine learning and crowdsourcing, making common features of websites accessible via commands to support a user's intent on a given website.

Figure 2 shows the system's user interface. Users can use the system once arriving at the website and pressing a shortcut to bring up the command prompt. Then, they specify what their intent is—the task they are looking to accomplish or feature they wish to access on that website. Users can directly enter the desired feature (i.e. their intent) and be taken directly to the page of interest. For example, to reach the registration page on the present website, users enter "sign up" as their intent, after which the system takes them directly to the page where they can complete the intended task. The purpose of our system is to offer consistent access to common features of websites. In addition to actively soliciting a specific feature, users can access a list of features supported by the website—as recognized by the system and identified by other users—available as "intent examples," then choose the one they wish to access. This list (Figure 2, right) surfaces the main features of the website that are supported by the system or have been previously identified by other users. This short list can help users quickly find links to common tasks on unfamiliar websites, such as

tracking a package or contacting the website owner, thus removing the need for users to "figure out" how these features were implemented for a particular website. We named our system prototype *CrowdIntent*, which was the name used for the system in the user study.

Associations between user intents/commands and target pages are done in three ways. The first one is by classifying hyperlinks on the website via trained machine learning models and assigning them to the initial system-supported intents. The second is by using pages marked as correct by users for previously entered intents, via majority voting. If machine learning fails and there are no previously identified pages for the user's intent, then the third mechanism is to perform a "feeling-lucky" background search for the given intent and domain (on DuckDuckGo). A "feeling-lucky" search redirects the user to the first entry in the search results. If none of these options work, users can ask for human input, which triggers a help request broadcast to online users on the website who are also using the system, asking them to find the page for the intent entered by another user. Help requests can also be attended to asynchronously, available through a list of unfulfilled requests at any given time. We refer to the annotations and real-time help as "human input."

3.1 Example Scenario

Consider a scenario where a blind user is looking to sign up for a specific grocery delivery service for which there is a subscription. Without our system, one of the ways to do so involves going to a search engine and entering appropriate keywords, then browsing through the search results using screen reader shortcuts or iterating through the result links. Another way is to find the website via search engines or enter the domain directly, then find the hyperlink on the website worded "pricing," "costs," or "subscription" via screen reader shortcuts or content search (e.g., Ctrl+F) or iterating through all links on the web page. Using our system, the user can simply go to any page on the grocery delivery website, then enter "pricing," "subscription," or a related synonym into the system, and they would be taken directly to the target page, which had been found either via machine learning or prior human input. Users are not required to know or memorize the commands ahead of time: they can use the list of features depicted in Figure 2 (right) to see the supported features for the websites they visit, identified automatically by the system or via human input. Users also do not have to provide a matching keyword for the specific hyperlink on the website that leads to the feature they wish to access. Instead, the system embeds synonyms into the command interpreter to identify the most likely feature from the command provided, and individual differences in implementation are handled by the patterns learned via machine learning, which is able to categorize "sign up," "register," and "create new account" as hyperlinks relating to the same high-level feature of different websites due to having been trained about a diverse set of such links.

3.2 Implementation

3.2.1 System Architecture. Figure 3 describes the system workflow. Once target pages are predicted and marked as correct, they are stored in the intent repository and become available for all users who enter the same intent in the future. The initial list of intents supported grows as pages are marked as correct for intents entered by users. Initially, available intents are those supported by the machine learning models we created. In other words, these are commands/features for which supervised learning models attempt to classify hyperlinks on the website as associated with each feature, which are listed as follows:

- Log in
- Reset password
- Sign up
- Help
- Contact
- Find deals
- Browse items
- Find popular items
- Pricing
- Find store
- Track order
- Search website

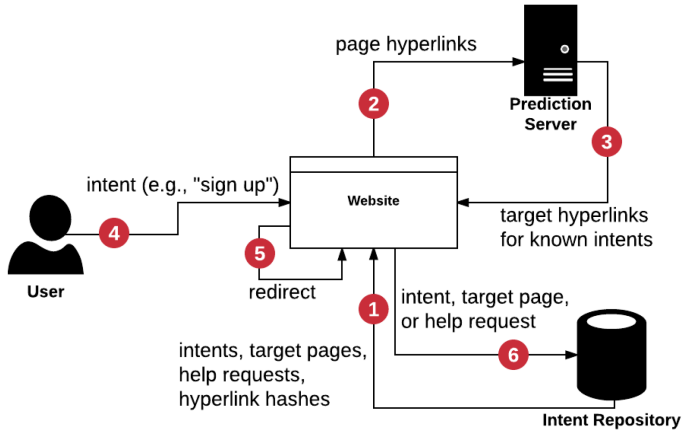

Fig. 3. System workflow. The browser connects to the intent repository and prediction server (via a browser extension). (1) The browser reads list of intents (initially supported), target pages, and hyperlink hashes from the intent repository, then (2) sends hyperlinks on current page to the prediction server, which will (3) return target hyperlinks for known intents. When users (4) enter an intent, it is parsed and interpreted, then (5) the browser extension checks for a target page or hyperlink, and if a new target page or hyperlink is identified either via machine learning, human input, or background search, (6) the data is sent to the intent repository after the user is redirected. The intent repository also manages requests for human input.

Users can specify these intents from any web page on the website. The reason we chose these intents is twofold. First, even though browser built-in password managers remember user credentials, users with visual impairments have great difficulty performing authentication tasks on websites [23], especially locating login, sign-up, and password recovery pages [12], due to the fact that these pages are named differently from one website to another (e.g., "sign in" vs. "log in," "sign up" vs. "register"). Second, the preceding and other features (e.g., "contact," "browse items," "help") are common, expected features in most websites, and like the authentication features, they also vary in how they are implemented. Therefore, these website-level features are good candidates to test command-based browsing for main website features.

When design changes are made to a website, such as changes to hyperlinks and URLs, the machine learning models might be able to readily detect the new hyperlinks for the intents supported, whereas the user might have to re-learn how to access the feature.

If a design change affects a target page associated via human input (instead of machine learning) such as the "buy gift card" command leading to a broken link, the user will mark the page incorrect, and upon finding the new, functioning target page, the new association will have precedence over the ones marked as incorrect (i.e., wisdom of the crowd).

Any command–target page association marked correct by users is available for future use by the same user and other users of the same website. Intents are defined with their corresponding input and a few synonyms configured in the back-end (e.g., "log in," "sign in," "login"), and when intents are entered, the given command input is checked against the list of supported commands (and their synonyms) and the command is mapped to the intent with the lowest Jaro-Winkler distance [73] (a string similarity metric) from the user input to the supported commands (or any of their synonyms). We implemented our system via user scripts that could be added to users' browsers using a browser extension such as TamperMonkey [68], making our system widely compatible.

Our proposed design considers scalability of supported commands in three major ways. The first one is via the system learning about a hyperlink that gets classified by the supervised learning

Table 1. Features in the Supervised Learning Model for Hyperlink-Based Website Feature Classification

Hyperlink: my account		
Feature	**Example**	**Transformation**
Text	*my account*	Vector of token occurrences in hyperlink's inner text
Context	*post to classifieds my account*	TF-IDF vector of hyperlink grandparent's inner text
Position	*0.1123*	The element's relative index in the document tree (%)
Length	*10*	Number of characters in the inner text
URL words	*log in home*	TF-IDF vector of URL word segmentation

models created for intents originally supported by the system. In such a case, if a link exists that maps to one of the main features supported, it will be immediately available for users to access via a command (Figure 2, left) or the list of supported features (Figure 2, right). The second way in which scalability is supported is by allowing users to manually mark a hyperlink they have found themselves, via clicking through the website to land on the target page, as a result of the background search query, or as a result of helping another user. This is the process that gets executed automatically when the machine learning models cannot find a corresponding hyperlink for a given command. In such a case, the new command is added to the list of supported commands, and any user who enters that command for that particular website will be redirected accordingly. This process will naturally surface important commands for each website over time with system usage, creating a long tail of shared, crowdsourced list of important commands for a given website (e.g., Figure 8 in the appendix shows all of the different commands entered by users during the study). The third way in which scalability is considered is via distant supervision (described in Appendix A), through which machine learning support for new intents can be added by the system developers.

3.2.2 User Workflow. From the perspective of the user, as soon as the page loads, the system starts working in the background to detect hyperlinks for supported intents (e.g., common features of websites) based on hyperlinks on the current page. When hyperlinks are classified, they are automatically mapped to the intents supported by the system. To bring up the system, users press a shortcut (e.g., pressing the Shift key twice), which prompts them for what they are looking to do (i.e., what their intent is). After they enter their intents, if there is a predicted hyperlink for the desired common feature, then users are directed to the target page via a simulated click on the classified hyperlink. If there is no predicted target page, users are redirected to the first search result of a "feeling-lucky" background search query constrained to the present domain. Users can mark pages as correct or incorrect for the intent, and such markings are considered later by the system in selecting the target page (e.g., select the page with the most correct marks), effectively leveraging the wisdom of the crowd [74]. If users believe the target page is incorrect, they can request human help—other users of the same website—to find and mark the page in real time, or later when someone is online. In our study, one of the researchers provided the real-time help to participants as if they were another user on the same website for a couple of tasks where this aspect was tested.

3.2.3 Website Feature Classification. We built **Support Vector Machine (SVM)** back-end models using scikit-learn [57] to classify hyperlinks leading to target pages for user intents. We used hyperlinks as the unit of classification following recommendations from prior works in web search and genre classification of web pages [6, 19, 27, 33, 34, 38, 41, 54, 55, 72]. We used features such as the text of hyperlinks, along with their grandparent's text in the document tree as their context, their relative index in the document tree, and word segmentation of the URL (example in Table 1).

We chose these features because they carry information about the hyperlink or button that gives access to a certain page, which has been found to be an efficient way to classify web pages [33].

We used a combination of data collected via distant supervision [45] (i.e., weakly labeling via heuristics) and manual labeling on **Amazon Mechanical Turk (AMT)**. We collected hyperlinks and screenshots for labeling by scraping a stratified sample of Alexa's Top 1 Million Websites list [4] as well as by looking at top five websites for each category on SimilarWeb [59], identifying common features across the categories and collecting similar websites via the *"related:"* search feature on Google for each of the top websites in each of categories.

Our resulting machine learning models have F1 validation (10-fold) scores ranging from 0.74 to 0.95, and 0.60 to 0.92 when making predictions on a hold-out test dataset containing a sample of 20% of instances not included during training. Model details are shown in depth in Appendix A.

3.2.4 Human Input. To mitigate machine learning failures and explore the potential of crowdsourcing in accessibility (e.g., [13, 28]) in our work, we added a help feature to our functional system. Users can explicitly ask for help when they determine the target page is incorrect for the intents they enter. If other users are online using the same website (and using the system), they receive a notification saying *"someone needs help finding the page for [user's entered intent]."* If users on the receiving side of the notification choose to help, they are asked to find the target page on the website and mark the page as correct when found, which will trigger a notification to the requesters saying that *"someone found the page for [user's entered intent],"* asking users if they want to visit the page, and taking the requesters to the page found by the helpers, if they choose to do so. Any page marked as correct by users is mapped to an intent and identified as "found by the crowd," becoming available to other users of the website in the future.

Human contributions are also provided passively, via marking pages found via machine learning or background search as "correct"—a feedback loop. After entering an intent and being taken to the page via either of these mechanisms, if the page is correct, users can mark the page as correct. If not, users can mark the page as incorrect, and they will be given the option to try to find the page manually or get human help (in the manner described earlier). In the former case (i.e., if the page is correct), user markings will be considered when intents are entered in the future by any user (including themselves) and the target page with the highest number of correct marks will be selected. In the latter case (when pages are marked as incorrect), the target page is moved down the list of intent examples because it was marked as incorrect before, with highest correct marks moving to the top of the list within the alphabetical order of intents. In addition, when target pages are marked as correct, a hash of the hyperlink or the actual page URL (if a hyperlink is not available, e.g., found manually or via web search) is stored in the intent repository so that in the future, hyperlinks matching the hash for the target page no longer need to be sent out to the server for classification, thus preventing unnecessary network traffic and computation. Figure 4 shows the human input workflow from the user's (the requester) perspective.

4 METHOD

Our goal was to study whether command-based browsing for main website features could improve the browsing experiences of users with visual impairments over their conventional/usual way of browsing websites. Our study aimed at evaluating the command-based paradigm focused on usability evaluation and insights from using a real system. We conducted the study with 15 participants, including 9 with visual impairments and 6 without. The reason we included users without visual impairments in our study was to gauge whether command-based navigation could also benefit users without visual impairments (i.e., the potential for universal benefit). In the system evaluation, we also included some tasks involving human input, which was evaluated to gauge

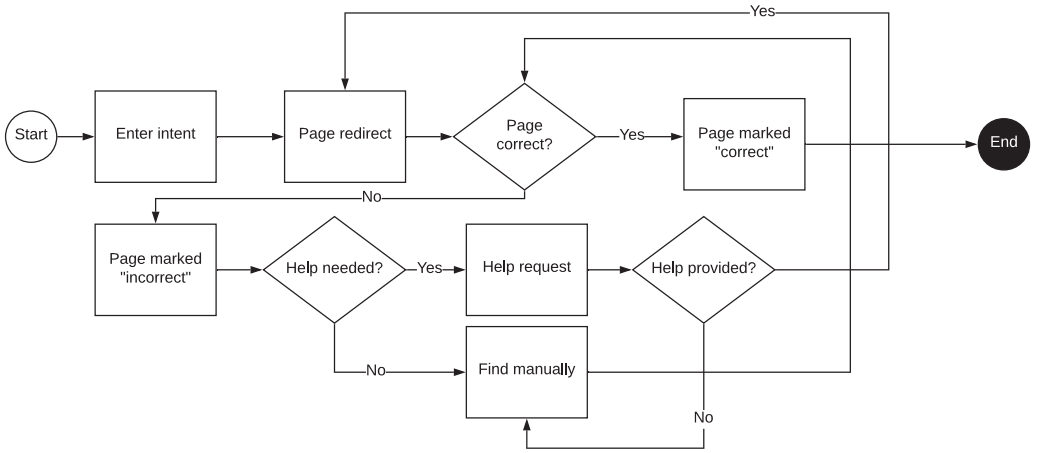

Fig. 4. Flow diagram of how human contributions are marked in the human input process. Contributions can be made by marking a target page as correct or incorrect after entering a command. Target pages can be found automatically via the system, via helpers who would choose to take on help requests, or manually by the user. Help requests can be made by users when they determine the page they were redirected to may not be correct for their entered intent/command.

user impressions of crowdsourcing access to main website features. We conducted our study in a lab setting between August 2017 and August 2018.

We conducted an evaluation of the overall system performance, such as the **System Usability Scale (SUS)** and task completion times from log data. We also incorporated a few exploratory components to this phase, namely gauging user acceptance and utility of crowdsourcing access to main website features, and a comparison with search engines (e.g., Google) in follow-up sessions with participants.

The system evaluation was preceded by a formative evaluation with 14 participants, conducted in the United States in the fall of 2016. The goal of this formative evaluation was to learn whether the command-based paradigm would be useful, acceptable, and desired by users with visual impairments, which we did through testing a Wizard-of-Oz [20] prototype along with participant interviews. This formative study aimed at understanding acceptance and perceptions of command-based browsing in general (e.g., the interaction modality) as well as the need for such an approach (e.g., what features to support), by hard-coding intents to target pages related to authentication on participant-provided websites ahead of user study sessions. Follow-up sessions helped uncover learning effects and whether participants preferred using the command-based paradigm after two sessions. We studied 14 participants in the formative study, including 7 users with visual impairments and 7 without. Table 2 shows a summary of the questions explored in the formative and summative evaluations.

4.1 Study Design

Our study involved observing and logging participants testing the system and their own conventional method of browsing, which we refer to as "methods" hereafter. Each participant tested both methods (i.e., within-subject, using the same websites for both methods in the session). To mitigate carry-over biases, we counterbalanced the order in which these two methods were tested; odd-numbered participants (e.g., P1, P3, P5, and so on) tested the conventional method first, whereas even-numbered participants (e.g., P2, P4, P6, and so on) tested the system first. Our participants included blind users, users with low vision, and users without visual impairments. Participants were

Table 2. Research Questions in the Formative Evaluation and the Evaluation of the Functional System

	Formative Evaluation (n = 14)	Summative Evaluation (n = 15)
Session 1	Do users like command-based browsing? What are some desired commands?	How beneficial is command-based browsing in practice? Is crowdsourcing commands a promising direction?
Session 2	Do users remember how to use commands and the system? Do users prefer using the system?	How does command-based browsing compare to using search engines? Are website-specific intents beneficial?

paid $30 for each session, lasting approximately 2 hours at their preferred location (e.g., their home, workplace, or our lab). To prevent social desirability bias, we did not voluntarily tell participants we built the system they were testing.

We encouraged participants to use their own computer to strengthen the ecological validity of our evaluation (i.e., participants used their usual computer, operating system, browser, and assistive technology). Otherwise, we provided a laptop with popular assistive technology (e.g., ZoomText, JAWS) and browsers (e.g., Chrome, Firefox, Internet Explorer) for participants to use. Our study was reviewed and approved by the institutional review board at Syracuse University.[1]

4.2 Study Procedure

This section describes our study protocol as experienced by participants. Participants started with some time for familiarization, then tested tasks, responded to a SUS questionnaire, and answered interview questions, twice (one for each method).

4.2.1 Familiarization. Participants started each session by completing three training tasks for each method on a familiar website to allow them to get acquainted with the methods. Participants could ask questions and then continued the study once they understood the methods. For all participants, the familiarization tasks were the same for both when using the system and when not: to find the page for logging in, resetting password, and buying a gift card on Amazon.com. In other words, participants were given time to test these three tasks before starting the actual study tasks. We did not allow participants to test these three tasks on Amazon.com for the actual study tasks due to the risk of carry-over effects from familiarization. This familiarization was also included in the counterbalancing mechanism we used. In other words, if the first method to be tested was the system, participants familiarized with the system first. If the first method to be tested was their conventional way of browsing, they were asked to test these familiarization tasks with that method before starting the actual tasks.

4.2.2 Tasks. After familiarization, participants used the system and conventional browsing (counterbalanced order) to test different commands on different websites. In the first session, participants used four websites, of which they picked three (two familiar, one unfamiliar), with the fourth being a fixed website we chose to pose as an unfamiliar website (a DMV (Department of Motor Vehicles) website from a different state than that of the participants: California). In the follow-up, participants picked all four websites: two familiar and two unfamiliar. Table 3 shows the structure of the study with regard to tasks and websites tested, and Figure 7 (in the appendix) shows all websites tested in the study. We cleaned up the intent repository before every session to ensure consistency among participants. Even though participants picked different websites, they tested the same tasks on the different websites they picked for both methods (e.g., always the same

[1]The study was conducted when the authors were at Syracuse University.

Table 3. Websites and Tasks Tested for Each Website

Session #1				
	Task #1	**Task #2**	**Task #3**	**Task #4**
Website #1 (familiar)	Log in	Reset password	User choice	Provide help
Website #2 (familiar)	Contact	Help	User choice (helped)	Researcher choice
Website #3 (unfamiliar)	Sign up	Log in	Reset password	–
Website #4 (unfamiliar—DMV)	Sign up	Appointment	Register to vote	Change address

Session #2				
	Task #1	**Task #2**	**Task #3**	**Task #4**
Website #1 (familiar)	Log in	Reset password	User choice	Provide help
Website #2 (familiar, shopping)	Browse items	Find deals	User choice (helped)	Researcher choice
Website #3 (unfamiliar)	Find popular items	Pricing	Sign up	
Website #4 (unfamiliar)	Search	Find store	Contact	

Note: Website #2 Task #3 and Website #1 Task #4 involved human input. Researcher-choice tasks (Website #2 Task #4) were chosen by the interviewer on the spot, upon knowing which familiar website the participant chose to use. The study was a within-subject evaluation between participants' conventional way of browsing and the system.

tasks on the first, second, third, and fourth website). Participants tested commands supported by the real system's machine learning models (presented earlier in Section 3) as well as commands of their own choosing (Figure 8 in the appendix shows all commands tested). Two of the tasks involved crowdsourcing, one with the participant being the requester, and another with the participant being the helper. We acted as the helpers when participants were the requesters.

In designing the study, we attempted to create a balance between ecological validity and controlled settings. For example, we allowed participants to choose the websites and two of the tasks to strengthen the ecological validity of our study, since different participants picked different, random websites and tasks. However, we ensured that participants picked two familiar and two unfamiliar websites, and made the fourth website in session 1 the same across all participants.

All participants used four websites, testing up to four tasks on each website, according to the structure in Table 3. In both methods, testing a task consisted of arriving at the target or entry page of the desired website feature. We did not ask participants to fully complete the task (i.e., interact with page-level content) since that is outside of our scope and covered elsewhere (e.g., [8, 13, 44, 49–51]). Participants tested study tasks in the order specified in Table 3. For example, for the first website (chosen by the participant), participants tested logging in, resetting a password, any task of their choosing, and fulfilling a help request. For the fourth website, in the first session, participants tested signing up, scheduling an appointment, registering to vote, and changing an address.

For the first session, participants (except for P14 and P15) were instructed to start the task on the home page of each website rather than using a search engine (e.g., Google) for the tasks. Participants could still use websites' internal search as needed. The decision to have the website's home page as the entry point for each task was done to mitigate any bias that could be introduced by challenges users could have when navigating search engine results (see [5, 23, 37, 56, 69]). In the follow-up session, we allowed participants to use search engines to test the tasks when using the conventional method to learn whether our system would still outperform the conventional method even when using search engines, resulting in eight participants: six follow-ups in addition to P14 and P15's first sessions, using search engines in the conventional method.

4.2.3 SUS Questionnaire. After completing the tasks with each method, participants responded to a SUS questionnaire about the methods. When using the SUS questionnaire for the conventional method, we used the word "method" instead of "system" and told participants to consider the usual

way of doing things—using their own tools and strategies (e.g., screen reader shortcuts, search engines, magnifiers)—when responding for that method.

4.2.4 Exit Interview. After working through the tasks with both methods, participants answered exit interview questions aimed at capturing participants' acceptance, preferences, and perceptions of the system (e.g., *"which browsing mechanism do you prefer and why?" "what tasks did you find the system most useful for?"* and *"when did you find the system to be rather unnecessary?"*). We also included questions about preferences on target page identification (e.g., system vs. human input) as well as individual questions about any particular behaviors that stood out. For example, if a user voluntarily praised a certain feature of the system, or looked confused or lost, we asked follow-up questions. We note that our goal was not to use our system as a benchmark, but to understand how command-based browsing for main website features could benefit users with varied abilities.

4.3 Participants

We recruited participants from our prior studies involving users with visual impairments, also reaching out to new participants via email, phone, Craigslist ads, and using mailing lists of local organizations serving people with visual impairments. We also encouraged participants to refer more prospective participants to us (i.e., snowball sampling). We recruited five participants who are blind, four with low vision, and six without visual impairments. Six participants did follow-up sessions, which were conducted to confirm whether our system would still be useful when participants could use search engines (e.g., Google, Bing), and we reached saturation after eight sessions (i.e., the same findings were repeatedly observed across the sessions). Our participants were from diverse backgrounds, including students, consultants, a retired teacher, a production worker, an IT consultant, a college professor, and unemployed. Participants who self-described as low-vision reported varied conditions, including presbyopia, amblyopia, visual-motor integration, chronic vertigo, bilateral coloboma of the iris and retina, and cerebral palsy with legal blindness in one eye.

For the formative study, we recruited three blind users, four with low vision, and seven without visual impairments for both the first and the follow-up sessions. The procedure for the formative study sessions was nearly identical to the procedure of the sessions with the functional system. In the end, we studied a total of 27 participants, 14 of whom have visual impairments and 13 do not, conducting 49 study sessions when considering the formative evaluation. Table 5 (in the appendix) shows the demographics of our participants.

4.4 Data Analysis

We instrumented back-end logs for intents entered, tasks tested, and target page identification mechanisms used (e.g., predicted, human input, or background search), among other interactions, using the logs and video recordings to measure completion times. Completion times from the videos were measured from the moment the task was given to the participant to the moment they confirmed the correct page (verbally in the conventional method), marking the time to reach the website and the time after reaching the website. For our quantitative analyses of completion times, we did not include the time for tasks involving human input, as those tasks involved the researcher.

We transcribed the audio recordings to conduct iterative thematic analyses [18] on the exit interview questions. Our process consisted of first immersing ourselves in the transcribed data by reading and actively looking for meanings and patterns. Then, two researchers independently coded the data at sentence level via open coding, creating codes for recurring ideas and insights, meeting to discuss and converge into 29 codes. Then, the codes were grouped and nine themes were derived from the codes, which guided our results presented in Section 5. The themes are *"Inconsistency of Websites," "Search Engine Challenges," "System Is Streamlined/Direct," "Utility of*

Crowd Commands," "System as a Fallback Mechanism," "Human Input Is Useful, with Reservations," "Habit of Conventional Method," "System Is Like a Search Engine," and *"System Should Be More Personal."*

5 RESULTS

We first summarize the findings from the formative evaluation, then focus on findings from the study with the real system for the remainder of the section. Throughout this section, we use the acronyms B for blind participants, LV for participants with low vision, and WVI for participants without visual impairments, according to participants' own self-descriptions.

5.1 Formative Evaluation

We learned through our formative evaluation that command-based browsing is a desirable mechanism for users with visual impairments, with the main advantage being an added layer of consistency across websites, giving users direct access to common features in a consistent way. For instance, P5 (LV) noted, *"well, I don't have to think about how it works each time. If I know it's going to work with the sites that I use, I don't have to rely on my intuition and using everyone's different interface."* Command-based browsing was deemed most useful by users with visual impairments, but users without visual impairments would use it as a fallback mechanism when they were visiting unfamiliar websites or completing infrequent tasks (e.g., changing a password). For example, P12 (WVI) noted, *"only if login is the criteria, then I can go with any of the methods. But if you asked me to change my password [...], then I would prefer the system which you installed."*

After the follow-up session, the majority of participants (11 out of 14) preferred our system over using conventional browsing, with our system being preferred by six out of seven users with visual impairments. Participants wished the system worked on different websites, supported more intents, and supported intents for specific website categories (e.g., *"edit social media profile"* on social media websites, *"change payment methods"* on shopping websites).

Blind and low vision users completed tasks faster with the system, and provided higher SUS scores (mean=80.8, SD=16.5) for the system for the system comparing to the conventional way of browsing (mean=50.4, SD=19.9).

These findings strongly supported our design intuition that the command-based paradigm would be better suited for users with visual impairments when accessing common features of websites. Users with visual impairments also wished that features specific to certain websites were accessible in a command-based manner. We also learned that users preferred the system in the follow-up session more than in the first session (see Table 5 in the appendix), and deeming it more useful in the follow-up session, as indicated by the SUS values given by participants. This formative evaluation was beneficial in validating our design intuition and probing users for their needs before placing significant effort into designing a functional system.

5.2 Tasks Completed

A total of 585 tasks (59 distinct commands) were completed during our study, across 57 distinct websites. Out of the 585 tasks, 317 were completed with the system and 268 without it (the conventional method). Fewer tasks were completed with the conventional method because our protocol did not require tasks where participants could solicit human help in the conventional method. When using our system, 72% (228) of the tasks were completed for features supported by the machine learning models, out of which 51% (116 out of 228) were successfully supported by model predictions, 29% (67 out of 228) supported by feeling-lucky background search, and 1% being found manually by participants. Out of the 317 tasks completed with our system, 21% (65) tasks were completed via help requests, which were answered by the researchers providing human input.

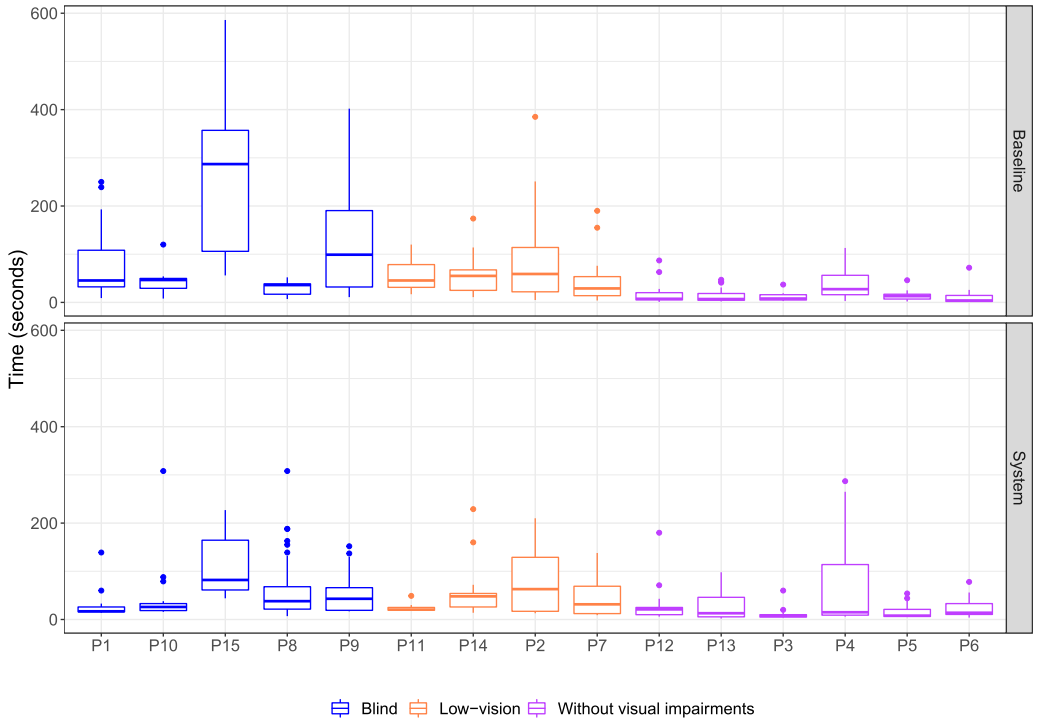

Fig. 5. Time measurements for all tasks in sessions 1 and 2. Baseline is the conventional method. We present per-participant measurements due to the small sample size and variability in participants' vision abilities. Differences were most notable for blind participants.

These results suggest that the machine learning models were effective in supporting the commands and features for which they were created. For a complete list of websites and tasks, see Figures 7 and 8 in the appendix.

5.3 Time to Complete Tasks

On average, task completion times were shorter with our system compared with the conventional browsing. Specifically, blind participants took an average of 42 seconds per task (median=31, SD=34.1) with our system and 112 seconds (median=51, SD=126) in the conventional method. Participants with low vision had an average completion time of 36.6 seconds (median=22, SD=33.3) with our system, and 61.5 (median=39, SD=65) using the conventional method. For participants without visual impairments, the average time with our system was 14 seconds (median=8, SD=14.1) and 19.5 seconds (median=12, SD=22.5) in the conventional method. These numbers support the idea that command-based browsing for main website features can reduce the effort users with visual impairments have to spend to access common website features. Figure 5 shows task completion times.

For participants without visual impairments, using the system generally contributed to slower task completion times, since they had to respond to and explore the system prompts shown on the screen, which was added overhead over dealing with the visual cues of each website. We note that time measurements for our system include the time waiting for the page-mark prompt, which took from 2 to 20 seconds, depending on the participant's preferences. This means that completion times for our system were actually shorter, but because we did not keep track of delay seconds

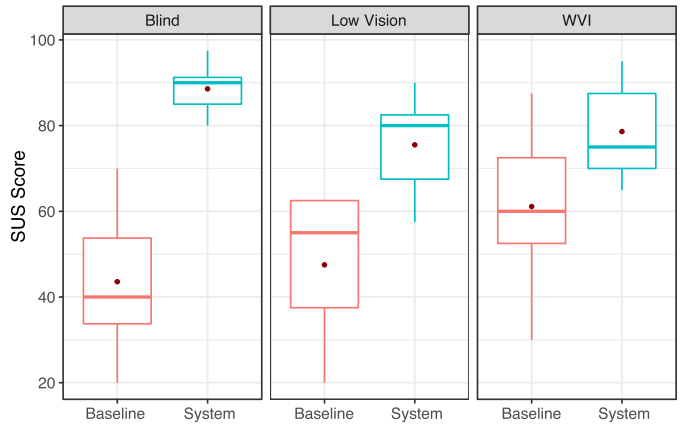

Fig. 6. SUS scores given by the different participant groups. Horizontal bars denote median, and points denote mean. Higher scores were given to our system, suggesting that participants perceived our system as easier to use.

used for each participant, we cannot subtract this number in our analyses. We set up this delay to accommodate for individual participants, since screen reader users took more time to determine the correctness of pages, whereas users without visual impairments found the delay unnecessary.

5.4 System Usability Scale

The average SUS scores given to our system was 81.2 (median=82.5, SD=11.2) and 52 to the conventional method (median=55, SD=18.8). The full SUS score is 100. For blind participants, the mean score for our system was 88.6 (median=90, SD=5.93), 75.8 (median=78.8, SD=11.6) for users with low vision, and 78.8 (median=75, SD=12.1) for users without visual impairments. For the conventional method, the average score of blind users was 43.6 (median=40, SD=17.1), 44.6 (median=46.2, SD=18) for low vision, and 65 (mean=61.2, SD=14.7) for users without visual impairments. Figure 6 shows the distribution of SUS scores by participant group.

In particular, participants with visual impairments gave average SUS scores of 88.6 (B) and 75.8 (LV) to our system, suggesting positive experiences. In comparison, their SUS scores of 43.6 (B) and 44.6 (LV) for the conventional method, which are well below 68 (the empirical average of large-scale studies) [10], indicated poor experiences. These scores further reinforce that our system was well received in enabling participants to access common website features more readily.

5.5 User Preference

In the first session, 12 out of 15 participants (80%, B=5, LV=3, WVI=4) preferred our system, whereas 3 (B=0, LV=1, WVI=2) preferred the conventional method. In the follow-up sessions, 4 out of 6 participants preferred our system (66.6%, B=2, LV=1, WVI=1), with 2 participants preferring the conventional method (B=0, LV=1, WVI=1). The following sections shed light on user preferences and impressions, as captured via our thematic analysis of the exit interviews.

5.6 Why Is Command-Based Browsing Preferred?

5.6.1 Consistency. The fact that there is a lot of inconsistency on the web was brought up by 11 participants, with the main consequence being having to figure out how to use each website, which can be overwhelming for users with visual impairments. For example, P8 (B) commented, *"since no two websites are the same, some place would say sign in, or log in, or whatever and with [the system] you just type in login and it puts you right where you need to be, therefore it is more consistent."* She

further added that such inconsistencies are aggravated by accessibility problems, saying *"I wish that people should check their sites to ensure that they are accessible to screen reader users. I think that would make things a lot easier for all of us instead of trying to figure out, like every website is a puzzle."* Her comments suggest that there is even more effort into "figuring out" websites when they are not designed and tested to work with assistive technologies used by people with visual impairments, which is a common occurrence. As a consequence, by incorporating a command-based paradigm for common and/or certain expected features of a website, a significant portion of this "figuring out" work could be avoided, at least for the most common features expected to be in most websites.

5.6.2 Fewer Steps and Challenges. Four other participants echoed P8's comments about accessibility, especially when navigating multiple steps to achieve their goals. For instance, P9 (B) noted, *"sometimes it's trial and error and it keeps changing, it's more visual and they are making it more sophisticated, the more it's sophisticated the more difficult for us. If I can get a page in 4 steps, tomorrow is different, tomorrow is 6 steps and it is very hidden."* P9's trial-and-error strategy was a common behavior observed during our study sessions, where participants would think a certain link would lead them to the desired feature, but they often had to go back or start over. He also noted that website changes can be challenging and make it difficult for him to access something he previously knew how to access. In this matter, command-based browsing can provide a consistent/expected way of accessing certain features, not requiring users to memorize the steps or pathways that lead them to a certain feature, and possibly not subject to fail due to changes in the website structure.

5.7 When Is Command-Based Browsing Most Useful?

As evidenced by participant preferences, most users with visual impairments preferred the system over the conventional method overall, meaning that they would use the system as a tool to help them complete tasks online because it is a more direct, consistent, and streamlined way of browsing websites. Participants with visual impairments have mentioned that there are *"very few steps,"* and that it is *"straightforward."* The following are the circumstances where command-based browsing was deemed *most* useful by participants in all groups, with and without visual impairments.

5.7.1 Unfamiliar, Cluttered, and Infrequently Accessed Websites. Command-based browsing was deemed most useful for websites that are unfamiliar, cluttered, or infrequently visited, and for features that are not frequently accessed, or features that are available across many websites. This was even the case for users without visual impairments. For example, P3 (WVI) found the system most useful *"for finding things that almost every website has that aren't immediately obvious to the eye like help and contact."* She explained her comment saying that although most websites have these features, they are implemented in different ways and many times she thinks they are purposefully "hidden" or "buried deep down" somewhere on the website structure, possibly to make it harder for people to access, such as contact information. This remark suggests that making common features more readily available could benefit many users, but it could also pose changes to the frequency in which certain features get accessed, which could be potentially unwelcoming to some website owners. For instance, because people find the contact feature more easily, the volume of requests for contact may increase.

5.7.2 Website-Specific Features. Giving users easier access to features via commands could also be helpful for website-specific features. For example, P10 (B) described his difficulty trying to change a setting on the bank website, when he was asked what commands he would like to use: *"Setting changes, and then again this goes back to account specific things. Like today I spent 5 minutes on Key Bank's website just trying to figure out how to change paper statements off so I was only getting*

electronic statements. It took me forever to try to find out where to go. And then I couldn't get JAWS to do it correctly because their web page is horrible." His comment reflects a common challenge of users with visual impairments, which is having to learn how the website implements a certain feature while dealing with accessibility problems along the way. We note that this may even happen on websites users are familiar with, but when the feature they wish to access is infrequently accessed.

5.8 Drawbacks

Participants also mentioned some drawbacks of using the system over the conventional method.

5.8.1 Discoverability. Two participants pointed out that by using direct input of intents instead of exploring and learning how to use each website, they are less likely to learn about features or links they are not aware of, which may help them complete another task in the future. For instance, P10 (B) noted discoverability as an advantage of the conventional method, saying *"pros, you kind of get a feel for how the web page is laid out. So if you're looking for something different next time you might already have an idea of where to find it or as you're browsing links to try to find something you might find something else that you didn't know was there."* Although the issue of discoverability was also identified previously in studies of personal assistants for users with visual impairments [1, 46], our findings also indicate that one drawback of command-based browsing is the potential for users to never become fully familiar with the features a website may offer, as their interactions will be more direct and focused on the desired features they wish to access at a given time. Nonetheless, our system can provide a list of supported intents on a website, although such a list would still be constrained by the intents supported by the system and intents mapped by the crowd.

5.8.2 Handling Failure. Participants raised concerns about human help availability (eight participants) and when machine learning does not find the page (four participants). For instance, P13 (WVI) noted, *"so if there's a new website, and no one knows about it and no one tried it, it's not going to help you."* Other concerns were related to whether people would help without incentives, and whether there would be anyone online: *"What if no one's there? I know you can't always be there. It depends on what time of day or night it is. Is this worldwide or is it local?"*

In regard to when the system fails, P2 (LV) noted, *"the cons are that sometimes the system did not work when you did a search to find something because it was not available on the website."* This is interesting because in contrast with keyword-based approaches, where either there are matches or there are not, it is not easy to tell when a machine learning model has failed, for example. This brings about the issue of trust and how heavily it depends on whether a personal assistant is able to fulfill the user request [1]. Nevertheless, 10 participants commented that having the help feature when the system fails is nice, and they would like to have this fallback option.

5.9 Comparison with Search Engines

Consistent with prior work [23, 69], we observed search engines being challenging for participants with visual impairments due to reasons such as difficulty while switching websites (e.g., back and forth between search engine and search result), trying to formulate and complete their search queries, and ultimately struggling to navigate search results provided by search engines.

5.9.1 Website Switching. The first reason given was the challenge of switching back and forth between websites. For instance, P15 (B) mentioned, *"[with the system] I don't need to worry about re-doing a search, like going back, exiting out of the whole search and then going back to, in this case Internet Explorer and it's much less frustrating."* She added that in some cases she was *"caught in a loop"* trying to find the page to complete the task. We observed that blind participants encountered this common challenge, which greatly hindered their experiences with search engines and caused

much frustration. There were many times during our study where participants searched for a feature associated with a website, and had to either go back to the search engine results or start their search over because they chose a link from the search results on a website that was different from the intended one, or a page on the intended website that did not implement the feature they wanted. When doing so, participants struggled to identify whether the browser had gone back, and where they were at that point in time (e.g., search results, the target website, or the search engine home page). Therefore, in supporting certain features via commands, a user would not have to deal with website switching, and instead could simply go to any page on the target website and provide a known command to access the desired feature. Arguably, website switching could also occur with the system if participants land on a different website when trying to complete a command, but the system would pick up where the user left off if they manage to go back to any page on the original website and the user would be able to continue their task there.

5.9.2 Keyword Matching Expectation. The second reason search engines can be challenging is because of the expectation that the results may depend heavily on the quality of the search query itself, which may lead to multiple trial-and-error attempts, and ultimately more steps. For example, P2 (LV) commented, *"because Google only uses keywords and then they combine all the keywords and then they can come up with something totally different than what you were looking for."* Her comment highlighted the fact that because certain features are implemented differently from one website to another, the keywords used in a search query therefore must somewhat match the implementation. Such a problem could be mitigated via commands that "learn" how to identify certain common features and access them, which would be more desired by users. For instance, after describing to participants how the system connects intents to website features at the end of the session, five participants commented that they would prefer it over keyword matching because they do not have to worry about formulating queries correctly. For instance, P5 (WVI) mentioned, *"so just knowing that it's a mapping thing and it knows the website structure and it's more likely to find the map and less likely to get deterred by a wrong keyword search, so when I was worried about having the exact word, I didn't really have to worry about it because it wasn't about a word, it was about a mapping function, so yeah, that makes me more confident."* This is an advantage over having to formulate a search query with the proper keywords that match a given website's implementation, but it also relies on the system to cover enough synonyms for each command. Although modern search engines do not merely match keywords, users may still have the expectation that their search query has great impact in the quality of the search results returned, and this can affect the user's confidence in finding what they need: five participants noted being careful to enter the "right" keywords into the system when entering commands.

5.9.3 Navigating Search Results. Navigating search results can be a cumbersome task for users with visual impairments. For example, P9 mentioned that he spends a long time browsing through search results and finding ways to determine which link is the most appropriate: *"I always try to use a shortcut, maybe initial if I can start, maybe, like that, you know,"* referring to how he navigates search results using JAWS' links list feature. His comment highlights the fact that in using search engines to access a desired feature of a website, users not only have to deal with the website itself but also the search engine website and the navigation between the two.

5.9.4 Search Engine Time Logs. In our study, eight participants used search engines, with seven of them being participants with visual impairments. Our time measurements among these users corroborate the challenges described by participants when navigating search results: the average time spent using search engines to access target website features was 90.9 seconds for blind participants (median=60.6, SD=119), 36.5 seconds for participants with low vision (median=23, SD=36.2),

and 20.9 seconds (median=18, SD=16.7) for the user without visual impairment. Among the seven users with visual impairments (B and LV), the average time to complete tasks with the baseline method was 166 seconds (median=114, SD=146) and 91.5 seconds (median=67.5, SD=57.2) when using the system. This means that even when using search engines, users with visual impairments spent less time finding the desired website features when they could use the system.

The breakdown of measured time between browsing search engine results and browsing the target website provides more nuance about this difference. For example, when considering only the time spent browsing search engine results to find the desired feature of a website, participants with visual impairments took an average of 99.6 seconds (median=67, SD=120) in the baseline, whereas the same measurement was 34 seconds (median=21, SD=31.6) with the system. The main reason for the shorter time with the system was that without it, participants attempted to find the correct website *and* target page combination for the desired feature by browsing the search engine results, often having to go back and forth, whereas with the system, participants did not have to worry about finding the correct page from the search results and instead only needed to find *any* page on the target website, then use the system to access the desired feature once at the website.

When considering proportions of time spent on the search engine and the website, among users with visual impairments, the average proportion *search engine:website* time—that is, the proportion of time browsing search engine results to the time browsing the target website—was 2.19 when using search engines as the baseline, with the same measurement being 0.73 for the system. In other words, when not using the system, participants with visual impairments spent more than double the time browsing search engine results than they did browsing the target website, which further heightens the challenges of navigating search results to find the desired feature of the target website. Accordingly, the time browsing the target website spent by users with visual impairments after clicking a search result was 66.6 seconds (median=39, SD=58.7) in the baseline and 57.4 (median=42.5, SD=43.4) with the system. This means that after reaching the website, the time taken to complete the task with and without the system was comparable, suggesting that the bulk of the difference was observed in having to browse through search engine results to find the correct link when not using the system. Finally, among users with visual impairments, the average percentage of time browsing search results in relation to the total time to complete a task was 55.6% in the baseline, whereas that time was 36.3% with the system, since participants only had to reach the website via any search engine results, then enter their command.

5.10 Feedback about Human Input

In general, participants liked having the ability to ask for help. For example P6 (WVI) noted that in the conventional method, *"if the website is just not navigable, you can't just hit a button and ask someone for help."* However, concerns and insights have emerged regarding the ability to get help from others when accessing common features of websites via commands.

5.10.1 Crowd Favorites. Five participants thought one advantage of the human contributions is surfacing common intents for a website, seeing it as a *"way of seeing how other people have dealt with things."* (P8, B), which was an intended effect of the human input process. In other words, by making a list of supported features of a website readily available and giving access to such features in form of commands, users can quickly learn what features are available and supported by websites when visiting them for the first time (i.e., main features).

5.10.2 Privacy/Security Concerns. Five participants raised privacy concerns about the help requests. For instance, P3 (WVI) noted, *"[...] I don't know if that makes me anxious or not that I just joined this community that you know is the security there I guess? If I get help from someone do they know who I am? Is there anyway they could hack me?"* This was a common concern in our study,

with participants believing that others would have access to their data or identity, and saying they would be more comfortable if volunteers were somehow affiliated with the website.

5.10.3 Affinity Groups. Four participants expressed that they believed the help would be most beneficial if they could make use of affinity groups, such as coworkers, people with similar or different abilities (e.g., other blind users, users who are not blind), local versus global users, people affiliated with the website, and people speaking different languages. For instance, P3 (WVI) commented about using the system in a professional setting, saying *"it would be useful if I could identify a group of people I wanted to have as my personal cluster of people who I ask [...] if I could group them like ask group one ask group two I could have for different projects."* In another instance, P8 (B) mentioned that she would like to use help from people she knows, saying *"strangers don't know you and your habits and the things you are interested in,"* adding that for certain tasks it would be useful to get help from another blind user, saying *"if there is another blind person helping you, I mean they may be more familiar with the site or whatever, can find something quicker, or know where to look for something."* Blind participants also commented that getting help from people without visual impairments could be helpful for certain tasks requiring vision abilities, such as buying a product with desired visual features (e.g., color, shape).

5.11 A Personal Browsing Assistant Emerges

Participants made several suggestions on how to improve the system, with support for more sophisticated tasks, making it more personal, and adding voice input capabilities.

For example, four participants mentioned that they would like to use more sophisticated and specific tasks in the system, such as "shop women's apparel," and that supporting such tasks would be quite beneficial. Multi-step, more interactive tasks were also suggested. For example, P10 (B) mentioned, *"It would be cool if I could type in what I want and [the system] would maybe come back and say here's the top 10 matches based on what you typed. And just by clicking on it, it would add it to your cart."* Participants also wanted to be able to express their intents via voice. For instance, P12 (WVI) mentioned, *"for the [the system] I would recommend a voice version so somewhere people might not need to type so just say it and that takes [them] to the page."* P10 (B) also noted that the system would also be very useful on mobile devices, where it is much more difficult to navigate websites.

Three participants wanted the system to be more personal. For example, P3 (WVI) suggested the system to adopt a more human-like presentation, saying *"I would like it to be more inviting, like with a smiley face or something that makes it feel like that's my friend. That's my go to person right there."* In making the system work more like a personal assistant, four participants wanted it to integrate many websites at once. For example, P8 (B) suggested, *"Well, would it be possible to add a search feature on there to go to different website? Instead of having to go to Google, you could do a search of websites within the prompt."*

These comments point to opportunities to integrate command-based web browsing into voice assistants to assist users in completing a variety of tasks within and across websites, and that doing so could benefit large number of users. For example, by enabling a user with motor impairments to give a command to a website on their smartphone, unnecessary steps and challenges could be avoided. These could be possible had there been a standard way to access common website features.

6 DISCUSSION

Our findings suggest that command-based browsing for main website features can help users with visual impairments overcome challenges when accessing common features of websites by

providing consistency in how such features are accessed. We show through our work that machine learning can be used to support command-based browsing, and that there are opportunities to integrate human contributions (e.g., crowdsourcing) into such systems. Our findings also suggest that command-based browsing could also benefit people without visual impairments when they use unfamiliar and/or cluttered websites, as well as when accessing features they would infrequently use. In this section, we discuss the potential of command-based browsing and associated design challenges.

6.1 Consistency

As observed in our study, the main benefit offered by command-based browsing is that it removes the need of "figuring out" each website as far as what features or tasks they support. More specifically, it removes the need to learn, for each website, how it serves a common feature that is expected to be present, but likely to be different in implementation. This is achieved by providing consistency in the way features are accessed, in which regardless of how websites are designed, a command will attempt to give the user access to the desired feature. In our implementation, the work of "figuring out" was transferred to the machine learning models, which identify the mapping of hyperlinks to features. Overall, such consistency in how to access the common features of websites was appreciated by most of the users with visual impairments who participated in our study.

6.2 Self-Confidence

Comments made by participants also suggest that such consistency not only streamlines the way in which users can access website features but also gives users more confidence that they will be able to access the desired features, by entering the same command on each website. According to participants, this is a desired characteristic because very often they are not confident that they will be able to access certain features, unless they have memorized each step along the way and the website did not change, in which case they would have to resort to trial and error. Five participants also mentioned that with command-based browsing, they would not have to "worry" about formulating a search query with the right keywords, like when using a search engine.

Consistency and self-confidence have been observed before in an evaluation of a shopping assistant for users with visual impairments [61] that works across websites, and our work provides additional evidence of such benefits being observed when accessing higher-level website features via commands. Such a paradigm could give users with visual impairments affordances akin to visually scanning a page for visual cues of access to common features (e.g., log in at top right).

6.3 But Aren't Search Engines Enough?

It is a fact that search engines provide great benefit to many user groups. In addition, features such as Google Search Sitelinks [30] aim at *"helping users navigate"* a website and *"find shortcuts that will save users time and allow them to quickly find the information they're looking for"* [30]. Therefore, one could argue that Sitelinks are serving a similar purpose of the system we designed. However, throughout the follow-up sessions, we observed participants with visual impairments having difficulties using search engines to access the desired features for the websites they tested, even when the features were listed under Google Sitelinks. These difficulties came from navigating the search engine user interface and results, and switching websites or "going back and forth" to find the relevant page for the feature they wished to access, which was behavior also observed in prior works (e.g., [5, 23, 37, 56, 69]). We observed through our studies that command-based browsing could prevent this from happening, by, for example, giving users the ability to enter the desired feature they wish to access anywhere on a website, from any page they visit. This would

avoid the work of switching back and forth, re-doing search queries, and having to find the right link among search results.

The aforementioned back and forth requires users to be aware of what website they are on, which can be confusing to screen reader users, since they seldomly rely on visual cues. For example, users with low vision using a screen magnifier can quickly tell whether they have gone back to the search results by observing a difference in the color theme or overall visual aspect of the page they are exploring, but hardly screen reader users. Screen reader users often have to explore both websites by listening to content headers, lists of links, and other content features, and in doing so they cannot easily distinguish from one website to another. Moreover, when interacting with both websites in this manner, they have to deal not only with potential accessibility problems of the target website they wish to access a feature on but also issues on the search engine, such as unintentionally clicking promoted links or phishing/illegitimate websites, which did happen during our study with Bing, the default search engine of Internet Explorer, thus making matters worse.

6.4 Design Implications

We identified three major design directions in which command-based browsing could be further explored. We describe these next.

6.4.1 Personal Assistants and Interaction with Websites. The first direction is through better integration with existing IPAs, which have been shown to improve the lives of people with visual impairments [1, 2]. For example, we identified in our study that users would like commands to be more sophisticated, such as "shop women's apparel," as well as commands to support tasks that could cross website boundaries, such as comparing prices and obtaining information from different sources. We also identified user needs around commands for more content-focused tasks. For example, P10 (B) wanted to *"expand the tool"* to support *"describing a picture or a graph*," and P9 (B) wanted to be able to ask for human input in tasks that require describing visual features of products. These suggestions point to an opportunity for IPAs from companies such as Apple, Amazon, and Google to further incorporate support for website-related tasks more seamlessly. As of now, such assistants go only as far as making a web search and showing results, and this has been noted as insufficient in prior studies (e.g., [32]), and as a direction worth further exploration (e.g., NL2API: integrating web services into IPAs [65]). There could also be an opportunity for crowdsourcing to be incorporated in these personal assistants—even if via affinity groups or phone contacts—especially for tasks in which machine learning may not be sufficient. This overall design direction is further reinforced by participant comments indicating that they wanted the system to support voice commands and become a more personal tool.

6.4.2 Tasks over Content. The second design direction is to explore how to design assistive technologies and websites with user intents in mind. For example, browsing based on intents may allow users to skip navigational hurdles as well as avoid inconsistencies caused by different design choices of website developers. Users with visual impairments rely on assistive technologies to navigate from page to page, often facing inaccessible content and continually changing websites [69]. In contrast, task-level assistance in accessibility is more concerned with getting things done [41] rather than simply making content accessible. Therefore, our work has implications for both the design of assistive technologies and that of websites in general. First, our work demonstrates how user intents could be incorporated into assistive technologies, which as of now relies on describing page content rather than supporting user goals, leaving it up to users to "figure it out." We argue that state-of-the-art assistive technologies could leverage machine learning, crowdsourcing, and

even search engines to empower users to complete common tasks. Second, in designing websites, developers can help users by providing a list of tasks that can be accomplished on the website, thus promoting discoverability. Another alternative would be to develop web metadata standards that allow developers to map certain pages to high-level user goals pertinent to each website. Expecting the entire web to change is unrealistic, but such infrastructure could also be a consistent way to support user intents in search engines and personal assistants.

6.4.3 Crowdsourced Browsing. In implementing command-based browsing, crowdsourcing can be a useful feature to consider. We identified in our study that human input could be most beneficial with affinity groups—for instance, allowing a user with visual impairments to request help from other users with visual impairments for websites they often visit. In other cases, it may be beneficial to allow a user with visual impairment to request help from a user without visual impairment to get help with visual tasks. Last, but not least, it may be beneficial to allow users to request help from people they know, such as family members, friends, and coworkers, also known as "allies" [31]. Without affinity groups, we noticed privacy concerns around the help feature, with users reporting being more comfortable with either affinity groups or website-affiliated helpers.

6.5 Anticipated Design Challenges

6.5.1 Restricted Areas. In supporting user intents, a major challenge for search engines and personal assistants is supporting transactions within restricted areas. For example, to check bank account balances, users must first log in. This is not a new challenge to search engines, and realistic solutions may involve developers adding metadata to their pages to map user intents to pages as well as providing users with a list of transactions post-login. This is also a major challenge for supervised learning approaches aimed at predicting target pages, as it can be difficult to develop a web scraper to collect training data from restricted areas at a large scale.

6.5.2 Fulfilling a Task Completely. Rather than predicting user intents, we allowed users to enter their intent directly. What other ways exist to implement intent-oriented browsing? For example, simply directing users to target pages gives them more agency as opposed to programming by demonstration approaches, but there are more levels to completing tasks than just redirecting to pages. Perhaps adding more features such as intent classification based on usage behavior [22, 24, 35] and page-level interactions [9] could help.

6.5.3 Long-Term Intents. Another open challenge in this direction is how to support long-term intents, such as "getting a driver's license." In such cases, the relevance of target pages would change over time. Building on our approach, we could incorporate stages into system intents and allow the model to predict likely hyperlinks for each stage based on crowd-contributed and page-level interaction patterns [7, 28].

6.5.4 Automatic Support of New Tasks. Browser-based IPAs should be able to support tasks beyond those incorporated into the machine learning models. Crowdsourcing can contribute to growing the list of system-supported intents via continuous machine learning. For example, when target pages are marked for specific intents on many websites, the supervised model could automatically be retrained to support the new intent to predict pages for the same intent in the future. The list of supported intents could also grow by collecting data in a large scale for distant supervision, which can be effective and cost efficient.

6.5.5 Privacy Concerns. Users could have mixed feelings about privacy in receiving human help with their intents. Most thought they could be vulnerable, whereas others made up their mind

based on their understanding of the technology. For example, some users believed that they could not be hacked when the helper is trying to find the page for them, not completing the task on their behalf. Such concerns align with prior work on crowdsourcing showing that although privacy concerns may hinder adoption [17], they can be mitigated by making users aware of how systems operate [39], which in our case could include statements such as "help is anonymous" or "there's no direct connection between your computer and the helper's." Other solutions are affinity groups (e.g., workplace [40], friends, family), which also came up in our study, and reputation systems [67].

6.6 Limitations

6.6.1 Simulated Helpers. In our study, a researcher provided the human input and participants knew this. We understand that this affects the ecological validity of our results of this system's portion. In addition, although we did include tasks involving users providing help, our evaluation did not focus on crowd workers. Nonetheless, we believe to have captured user preferences about receiving human help to access main website features.

6.6.2 Introduction of New Risks. Although in our study it was assumed that users would be well intentioned, a new vulnerability is also introduced when an ill-intentioned user aims to deceive others via our system. Although phishing is mitigated by the imposed domain-binding requirement of our system, users could be misled to access pages they do not intend, for the potential benefit of the attacker. For example, one could offer to help only to point the user to a page promoting a product they are selling. One potential solution surfaced in our study, such as to enable users to identify trusted helpers, such as family and friends, which could mitigate risks for websites that may be target of attackers such as those involving involving payment or financial information.

6.6.3 Limited Number of Intents. Our system supports a limited number of intents via machine learning. Nonetheless, we showed that our approach of mapping target pages to intents could work well in providing easy access to features that are common across websites. Moreover, users did not know about the distinction between background search and machine learning being used to fulfill their commands. Therefore, we believe that user impressions about what the system can achieve went beyond the main website features originally supported by machine learning, due to the system having fulfilled specific commands users entered during the study (see Figure 8 in the appendix). Although the initial list of intents is limited, we believe that the design we propose can support scalability to more cross-website and website-specific commands as the system gets usage over time. That being said, we did not systematically evaluate the scalability of the system in this exploratory laboratory study, which could be done in future works.

6.6.4 Voice Commands Not Supported. The system only supported typed-in commands. The system could be implemented to support voice commands and be used on mobile devices with relative ease, therefore not being limited to desktop or laptop computers. Nevertheless, our decision to support typed-in commands on desktops and laptops was motivated by (1) the fact that users with visual impairments rely heavily on the keyboard for navigation [70, 71], and (2) that our intention was to evaluate the interaction modality of intent-oriented browsing as a formative step in the process of supporting this modality across other devices and platforms more broadly. We note however that (1) speech recognition remains a challenge for personal assistants [1, 46]; (2) users with visual impairments rely heavily on the keyboard for navigation [70, 71]; and (3) users with visual impairments are generally concerned about drawing unwanted attention when interacting with speech-based technologies, believing that such interactions can compromise their privacy through eavesdropping and surveillance [3].

7 CONCLUSION

Differences and inconsistencies in how websites are structured make browsing quite challenging for users with visual impairments, who often rely on linearly listening to content structure via a screen reader or magnifying small sections of the page. Such challenges could be overcome by incorporating a command-based paradigm that gives users access to main, common features of websites in a consistent manner. To address this problem, we iteratively designed and evaluated a system implementing a command-based paradigm to access main website features, implementing a layer of consistency for commonly supported features across websites via machine learning and human input. Our user study with 15 participants—including 9 with visual impairments—showing that command-based browsing can greatly improve the experiences of users with visual impairments by providing a layer of consistency through which they can access common features on different websites. People without visual impairments also found command-based browsing beneficial, but more so on unfamiliar, cluttered, or infrequently visited websites. We also found that search engines may be insufficient in supporting access to main, common features of websites and that human input could also be beneficial, especially via affinity groups. Our findings have implications for incorporating the design of command-based interactions into assistive technologies and integrating common website features into full-fledged IPAs (e.g., Siri, Alexa) to benefit a large number of users.

APPENDICES

A MACHINE LEARNING DETAILS

In this section, we provide an in-depth description of our machine learning approach to classifying hyperlinks associated with the website features supported by the system.

A.1 Classification Unit

In attempting to map website pages to main website features, we drew inspiration from prior works on retrieving search results for transactional search queries [41] and genre classification of web pages [6, 54, 55]. Hyperlink features are good indicators of what transactions the target page can support [34, 41], and web page genres can assist users in goal-directed browsing [29] as well as in improving effectiveness of web searches [53–55, 75]. Moreover, structural information (e.g., URL, anchor text, nearby text) can help improve link-based page classification [19, 27, 38, 72]. With fast link-based classification of pages surpassing accuracy of content-based classification [33], we deemed hyperlinks as practical indicators of target pages for user intents because the system would not be required to parse the content of potential target pages, but only consider hyperlinks leading to them instead. Therefore, our classification task consists of identifying the main website features based on their originating hyperlinks that point to the entry or target page for such features. For example, to support the command "find stores," the system inputs hyperlinks into a trained model that learned from many labeled instances of hyperlinks for the store-finding feature to determine if any hyperlink is classified for that feature.

To obtain our training dataset, we turned to the methodologies used in genre classification of web pages [6, 54, 55], except unlike prior work in this field, we label *origin hyperlinks* instead of *pages*. Still, in manually labeling the hyperlinks, what human annotators saw were the rendered target pages resulting from a click on the respective hyperlink that was being annotated. In other words, annotators saw (and annotated) screenshots of rendered web pages to choose a website feature that is present on that page, but in reality each screenshot was associated with an originating hyperlink that received the label. Our dataset contains weakly labeled data obtained through

distant supervision [45] (not done before in web genre labeling) and manually labeled data. Distant supervision consists of using heuristics to "weakly" label a training set [45]. This approach allowed us to obtain many training samples without prohibitive costs. Our target classes are "login," "register," "recover password," "help," "faq," "contact," "find deals," "browse items," "find popular items," "pricing," "find store," "track order," "search website," and "other."

A.2 Distant Supervision

We applied distant supervision by scraping websites that were sampled in two ways. In one approach, we knew ahead of time which features to support, and in the other, we identified features based on common features of websites in different categories (e.g., shopping, entertainment).

In the first approach, used for authentication and contact-related intents, we extracted a stratified sample of 3,000 websites from Alexa's [4] list of top 1 million websites in October 2016. We selected 1,000 websites from the top of the list, 1,000 from the bottom, and 1,000 random websites between the 1,001st and 999,999th website, inclusive. We then removed from our sample any websites that were not ".com," attempting to filter out most non-English websites. Finally, because we would manually label a sample of the data based on screenshots, we used a blocklist to remove any websites with adult and violent content, resulting in a sample of 1,486 websites.

Still on the first approach, we then defined our heuristics for distant supervision by visiting the top 100 websites on Alexa's Top 1 Million list manually and extracting the inner text of hyperlinks leading to the target pages of interest. For example, for the "login" class, we had keywords such as "sign in," "log in," "login," and "my account."

The second sampling approach involved obtaining popular websites via SimilarWeb [59] for each top-level website category (e.g. Food and Drink, Shopping) and identifying common features across websites from these categories. This approach was used for the various features that were not related to authentication or contact actions. We first collected the top five websites in each top category, then used the Google search engine with the *"related:"* search feature to obtain similar websites to each top website. This resulted in a list containing 519 distinct websites that were scraped using the heuristics defined in the following.

The heuristics used in the second approach were obtained by visiting each of the top five websites from each top category on SimilarWeb and collecting the inner text of hyperlinks leading to the common features across the websites (e.g., browse items, find deals, pricing, find popular items), which were later used for scraping.

The two preceding procedures resulted in heuristics used to scrape websites for hyperlink instances and label them automatically in the process. After gathering the heuristics, we developed and used a web scraper built with Scrapy [58] and Splash [60] to visit the websites present in our samples and collect the hyperlink features of the elements matching the keywords for each target class, thus labeling them automatically as belonging to the respective target class if they were found. We also collected screenshots from the page rendered immediately after the hyperlink was clicked, assigning each screenshot to its respective hyperlink. For every website, we also collected up to five random hyperlinks belonging to an "other" class (i.e., hyperlinks not belonging to any of the other target classes according to heuristics). This process resulted in the weak labeling of 11,931 hyperlinks.

A.3 Manual Labeling

From the weakly labeled dataset containing authentication (e.g., login, register, reset password) and contact-related hyperlinks (e.g., contact, FAQ, help), we selected a random sample of 688 authentication-related hyperlinks and a random sample of 688 contact-related hyperlinks for

manual labeling. In each sample, we made sure to have 50% of the samples belonging to the "other" class, as labeled by the web scraper. We conducted one crowdsourcing experiment (separately) for each sample on AMT. This was to ensure that crowd workers would not be overwhelmed with the multiple options for each page they had to label. We set up a Human Intelligence Task (HIT) consisting of labeling 10 pages. Before they could label the actual pages, in addition to being required to have over 95% all-time approval rates, crowd workers completed a qualification task in which they had to correctly label 7 out of 10 "training" samples after seeing four examples.

Labeling a page consisted of viewing a screenshot and choosing from a multiple choice question on which type of page (i.e., our target classes) users thought each to be, giving their confidence level ranging from 1 to 5, and entering the major reasons behind their decision into a text field. We followed recommendations set forth by prior work on user-based labeling of web page genres [6, 54]. We had each hyperlink labeled by five different crowd workers, taking the majority vote as the true class, and we did allow crowd workers to assign multiple classes to each page. However, we removed these multi-class instances from our dataset because there were fewer than 10 belonging to multiple target classes after majority voting. The experiments were conducted following a successful small pilot labeling 96 authentication-related hyperlinks on AMT.

All in all, a total of 774 unique crowd workers participated in our labeling experiments, each of whom were compensated with $0.04 per labeled page, with a total of $0.80 per task, which consisted of 10 qualification pages and 10 actual pages. The median time to complete the task was about 8.6 minutes. The inter-coder reliability score (Fleiss kappa) was 0.853 for authentication-related pages and 0.766 for pages not related to authentication (i.e., help and contact), which are considered acceptable agreement [6].

In the end, we obtained a dataset of 11,931 hyperlinks, with 1,454 (≈12%) being manually labeled and 10,477 weakly labeled (≈88%).

A.4 Models

We created three models—one for authentication-related pages, another for "help" and "contact" pages, and another for various tasks—based on an SVM classifier with linear kernel, combining all of the data obtained through distant supervision and manual labeling. In the system, we mapped the "get help" command to target pages classified as either "help" or "faq." We created three models because we conducted the scraping and manual labeling experiments separately for each subset of intents—for example, in the experiment with the "help" pages, there could be "login" pages marked as "other," which would confuse a single model. Table 4 shows the results of the performance evaluation of our models. The model is able to predict pages for the "recover password" (F1=.92) and "contact" (F1=.90) intents most accurately but making more mistakes on "track order" (F1=.57) and "faq" (F1=.75) hyperlinks. Coincidentally, the "track order" and "faq" had the smallest sample sizes within their feature groups, which could explain the poor performance, but could also suggest that the weak labeling approach was not effective for these two features. The models performed best when trained with both data obtained through distant supervision and manual labeling, but have reasonably accurate performance when training on weakly labeled data alone, as seen in the results involving hyperlinks for the *various* features (see the bottom section of Table 4). We created our models with scikit-learn [57] and used them in the back-end, available on a web server running a Python application, where an instance of each model was kept in memory for real-time predictions. Each model classifies all of the hyperlinks transferred from the browser, and in the end, a final list of classifications is merged and returned to the browser with the response containing the class (i.e., the main website feature) for each hyperlink.

Table 4. Model Evaluation of SVM Classifiers to Predict Target Hyperlinks for Common Features, Ordered by F-1 Score, with the "other" Class Always Last

10-Fold Cross Validation					Hold-out Manually Labeled Test Set				
Authentication Tasks					Authentication Tasks				
	Precision	Recall	F1	#		Precision	Recall	F1	#
recover password	.87	.94	.9	244	recover password	.94	.89	.92	19
login	.87	.91	.89	368	login	.86	.82	.84	22
register	.82	.83	.82	368	register	.9	.79	.84	24
other	.98	.97	.97	3,641	other	.88	.93	.9	19
				4,621					150
Contact & Help Tasks					Contact & Help Tasks				
	Precision	Recall	F1	#		Precision	Recall	F1	#
contact	.95	.95	.95	413	contact	.92	.88	.9	83
faq	.87	.87	.87	134	faq	.75	.75	.75	36
help	.75	.74	.74	168	help	.66	.55	.6	42
other	.99	.99	.99	3,548	other	.89	.95	.92	179
				4,263					340
Various Tasks					Various Tasks				
	Precision	Recall	F1	#		Precision	Recall	F1	#
pricing	.98	.87	.92	150	pricing	.88	.96	.92	24
find deals	.90	.91	.91	103	find deals	.73	.86	.79	22
track order	.88	.88	.88	40	track order	1	.4	.57	5
find popular items	.93	.8	.86	178	find popular items	.78	.97	.86	33
find store	.85	.8	.82	55	find store	.89	.84	.86	19
search	.85	.76	.8	252	search	.71	.87	.78	39
browse items	.85	.74	.8	244	browse items	.76	.79	.77	43
other	.94	.99	.97	2,025	other	.99	.94	.96	425
				3,047					610

Note: The hold-out sets contain a random sample of 20% of the manually labeled data, except for the "various tasks" model, in which there was no manually labeled data. The 10-fold Cross Validation column includes all of the data (both manually and weakly labeled). The prediction results on the test sets indicate that our approach can successfully assign hyperlinks to common website features.

In predicting hyperlinks in real time, we noticed the word segmentation algorithm was a performance bottleneck when handling query string values such as *"&pf_rd_r=MN30C7K3CER2BJNK1V72*," and therefore we excluded query string *values* from word segmentation, considering only URL path components and query string *keys* for the *url words* feature. The system also sends hyperlinks for prediction in three batches, with the first batch being a third of hyperlinks from the top of the page, the second being a third from the bottom, and the third being the ones in between. In doing so, target pages are available more quickly since website-wide hyperlinks—that is, common features of websites—are likely located at the top or at the bottom, with the middle portion most likely showing page-specific content.

B PARTICIPANT DEMOGRAPHICS

Table 5. Table of User Study Participants and Their Preferred Methods

ID	Gender	Age	Self-Reported Ability	Assist. Tech.	Prefer. S1	Prefer. S2
P1*	M	35	Blind	JAWS	**X**	
P2	F	56	Low vision (20/400 w/correction, legal blindness)	ZoomText	**X**	**X**
P3	F	65	Without visual impairments	None	**X**	
P4	M	72	Without visual impairments	None	**O**	**O**
P5	F	26	Without visual impairments	None	**X**	
P6	F	22	Without visual impairments	Magnifier	**O**	
P7	F	53	Low vision (legal blindness)	OS	**O**	**O**
P8*	F	47	Blind	JAWS	**X**	**X**
P9	M	45	Blind	JAWS	**X**	**X**
P10	M	42	Blind	JAWS	**X**	
P11	M	70	Low vision (no further description given)	ZoomText	**X**	
P12	M	30	Without visual impairments	None	**X**	
P13	M	24	Without visual impairments	None	**X**	**X**
P14	F	49	Low vision (legal blindness)	ZoomText	**X**	
P15	F	60	Blind	JAWS	**X**	
			Formative Evaluation			
ID	**Gender**	**Age**	**Self-Reported Ability**	**Assist. Tech.**	**Prefer. S1**	**Prefer. S2**
P1	M	60	Low vision (presbyopia, amblyopia)	None	**X**	**X**
P2	M	70	Blind	VoiceOver	**O**	**X**
P3	F	45	Blind	JAWS	**X**	**X**
P4	F	19	Low vision (visual-motor integration)	None	**X**	**X**
P5	M	25	Low vision (bilateral colobomas)	None	**X**	**X**
P6	F	52	Low vision (cerebral palsy, legal blindness)	None	**X**	**X**
P7	M	34	Without visual impairments	None	**O**	**O**
P8	M	33	Blind	JAWS	**X**	**X**
P9	M	25	Without visual impairments	None	**O**	**O**
P10	F	24	Without visual impairments	None	–	**X**
P11	M	26	Without visual impairments	None	–	–
P12	F	61	Without visual impairments	None	**X**	**X**
P13	M	72	Without visual impairments	None	**X**	**X**
P14	F	56	Without visual impairments	None	**X**	**X**

*P1 and P8 were also recruited in the formative evaluation. X, preferred our system; O, preferred conventional browsing; S1, first session; S2, follow-up session.

C WEBSITES USED AND TASKS COMPLETED

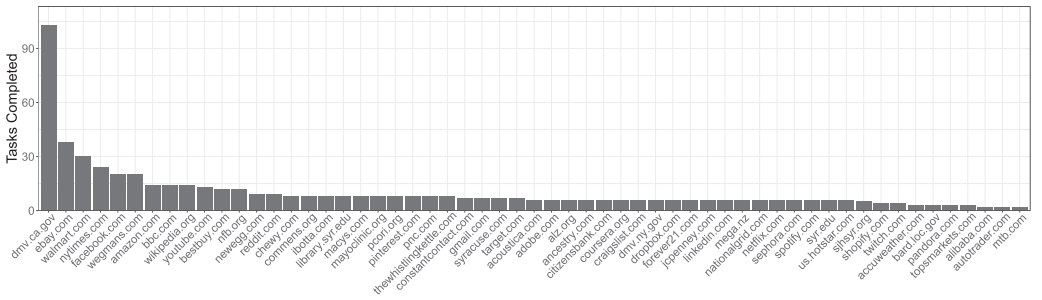

Fig. 7. Frequency of tasks completed by the website used in the study. A total of 57 distinct websites were used.

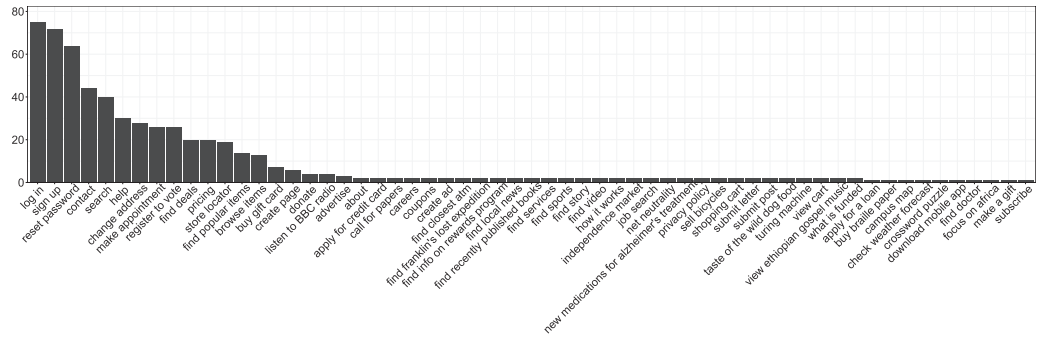

Fig. 8. Number of tasks completed per command.

ACKNOWLEDGMENTS

We thank our participants for sharing their insights. We also thank the anonymous reviewers for their feedback.

REFERENCES

[1] Ali Abdolrahmani, Ravi Kuber, and Stacy M. Branham. 2018. Siri talks at you: An empirical investigation of voice-activated personal assistant (VAPA) usage by individuals who are blind. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 249–258.

[2] Ali Abdolrahmani, Kevin M. Storer, Antony Rishin Mukkath Roy, Ravi Kuber, and Stacy M. Branham. 2020. Blind leading the sighted: Drawing design insights from blind users towards more productivity-oriented voice interfaces. *ACM Transactions on Accessible Computing* 12, 4 (2020), 1–35.

[3] Tousif Ahmed, Patrick Shaffer, Kay Connelly, David Crandall, and Apu Kapadia. 2016. Addressing physical safety, security, and privacy for people with visual impairments. In *Proceedings of the 12th Symposium on Usable Privacy and Security (SOUPS'16)*. 341–354.

[4] Alexa. 2017. Alexa. Retrieved March 18, 2022 from https://www.alexa.com.

[5] Patrizia Andronico, Marina Buzzi, Carlos Castillo, and Barbara Leporini. 2006. Improving search engine interfaces for blind users: A case study. *Universal Access in the Information Society* 5, 1 (2006), 23–40.

[6] Noushin Rezapour Asheghi, Serge Sharoff, and Katja Markert. 2016. Crowdsourcing for web genre annotation. *Language Resources and Evaluation* 50, 3 (2016), 603–641.

[7] Vikas Ashok, Syed Masum Billah, Yevgen Borodin, and I. V. Ramakrishnan. 2019. Auto-suggesting browsing actions for personalized web screen reading. In *Proceedings of the 27th ACM Conference on User Modeling, Adaptation, and Personalization*. 252–260.

[8] Vikas Ashok, Yevgen Borodin, Svetlana Stoyanchev, Yuri Puzis, and I. V. Ramakrishnan. 2014. Wizard-of-Oz evaluation of speech-driven web browsing interface for people with vision impairments. In *Proceedings of the 11th Web for All Conference*. ACM, New York, NY, 12.

[9] Vikas Ashok, Yury Puzis, Yevgen Borodin, and I. V. Ramakrishnan. 2017. Web screen reading automation assistance using semantic abstraction. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*. ACM, New York, NY, 407–418.

[10] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of Usability Studies* 4, 3 (2009), 114–123.

[11] Gabriel Barata, Tiago Guerreiro, and Daniel Gonçalves. 2009. *Blaze: Automating User Interaction in Graphical User Interfaces*. Ph.D. Dissertation. Instituto Superior Técnico.

[12] Natã M. Barbosa, Jordan Hayes, and Yang Wang. 2016. UniPass: Design and evaluation of a smart device-based password manager for visually impaired users. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, New York, NY, 49–60.

[13] Jeffrey P. Bigham, Tessa Lau, and Jeffrey Nichols. 2009. Trailblazer: Enabling blind users to blaze trails through the web. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*. ACM, New York, NY, 177–186.

[14] Yevgen Borodin. 2008. Automation of repetitive web browsing tasks with voice-enabled macros. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 307–308.

[15] Yevgen Borodin, Jeffrey P. Bigham, Glenn Dausch, and I. V. Ramakrishnan. 2010. More than meets the eye: A survey of screen-reader browsing strategies. In *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*. ACM, New York, NY, 13.
[16] Yevgen Borodin, Jeffrey P. Bigham, Rohit Raman, and I. V. Ramakrishnan. 2008. What's new? Making web page updates accessible. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*. 145–152.
[17] Erin Brady, Jeffrey P. Bigham, et al. 2015. Crowdsourcing accessibility: Human-powered access technologies. *Foundations and Trends® in Human–Computer Interaction* 8, 4 (2015), 273–372.
[18] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006), 77–101.
[19] Nick Craswell, David Hawking, and Stephen Robertson. 2001. Effective site finding using link anchor information. In *Proceedings of the 24th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 250–257.
[20] Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz studies–why and how. *Knowledge-based Systems* 6, 4 (1993), 258–266.
[21] T. B. Dinesh, S. Uskudarli, Subramanya Sastry, Deepti Aggarwal, and Venkatesh Choppella. 2012. Alipi: A framework for re-narrating web pages. In *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility*. 1–4.
[22] Alan Dix, Giorgos Lepouras, Akrivi Katifori, Costas Vassilakis, Tiziana Catarci, Antonella Poggi, Yannis Ioannidis, Miguel Mora, Ilias Daradimos, Nazihah Md Akim, et al. 2010. From the web of data to a world of action. *Web Semantics: Science, Services and Agents on the World Wide Web* 8, 4 (2010), 394–408.
[23] Bryan Dosono, Jordan Hayes, and Yang Wang. 2015. "I'm Stuck!": A contextual inquiry of people with visual impairments in authentication. In *SOUPS*. 151–168.
[24] Rob Ennals, Eric Brewer, Minos Garofalakis, Michael Shadle, and Prashant Gandhi. 2007. Intel mash maker: Join the web. *ACM SIGMOD Record* 36, 4 (2007), 27–33.
[25] Alexander Faaborg and Henry Lieberman. 2006. A goal-oriented web browser. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 751–760.
[26] Eelke Folmer, Bei Yuan, Dave Carr, and Manjari Sapre. 2009. TextSL: A command-based virtual world interface for the visually impaired. In *Proceedings of the 11th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 59–66.
[27] Johannes Fürnkranz. 1999. Exploiting structural information for text classification on the WWW. In *Proceedings of the International Symposium on Intelligent Data Analysis*. 487–497.
[28] Prathik Gadde and Davide Bolchini. 2014. From screen reading to aural glancing: Towards instant access to key page sections. In *Proceedings of the 16th International ACM SIGACCESS Conference on Computers and Accessibility*. 67–74.
[29] Aaron Genest, Carl Gutwin, Adrian Reetz, Regan Mandryk, David Pinelle, and Andre Doucette. 2009. Looking ahead: A comparison of page preview techniques for goal-directed web navigation. In *Proceedings of the IFIP Conference on Human-Computer Interaction*. 378–391.
[30] Google. 2020. Sitelinks. Retrieved March 18, 2022 from https://support.google.com/webmasters/answer/47334?hl=en.
[31] Jordan Hayes, Smirity Kaushik, Charlotte Emily Price, and Yang Wang. 2019. Cooperative privacy and security: Learning from people with visual impairments and their allies. In *Proceedings of the 15th Symposium on Usable Privacy and Security (SOUPS'19)*.
[32] Jiepu Jiang, Ahmed Hassan Awadallah, Rosie Jones, Umut Ozertem, Imed Zitouni, Ranjitha Gurunath Kulkarni, and Omar Zia Khan. 2015. Automatic online evaluation of intelligent assistants. In *Proceedings of the 24th International Conference on World Wide Web*. 506–516.
[33] Min-Yen Kan and Hoang Oanh Nguyen Thi. 2005. Fast webpage classification using URL features. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*. ACM, New York, NY, 325–326.
[34] In-Ho Kang. 2005. Transactional query identification in web search. In *Proceedings of the Asia Information Retrieval Symposium*. 221–232.
[35] Melanie Kellar and Carolyn Watters. 2006. Using web browser interactions to predict task. In *Proceedings of the 15th International Conference on World Wide Web*. ACM, New York, NY, 843–844.
[36] Melanie Kellar, Carolyn Watters, and Michael Shepherd. 2007. A field study characterizing web-based information-seeking tasks. *Journal of the American Society for Information Science and Technology* 58, 7 (2007), 999–1018.
[37] Friederike Kerkmann and Dirk Lewandowski. 2012. Accessibility of web search engines: Towards a deeper understanding of barriers for people with disabilities. *Library Review* 61, 8–9 (2012), 608–621.
[38] Wessel Kraaij, Thijs Westerveld, and Djoerd Hiemstra. 2002. The importance of prior probabilities for entry page search. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 27–34.

[39] Walter S. Lasecki, Phyo Thiha, Yu Zhong, Erin Brady, and Jeffrey P. Bigham. 2013. Answering visual questions with conversational crowd assistants. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 18.

[40] Gilly Leshed, Eben M. Haber, Tara Matthews, and Tessa Lau. 2008. CoScripter: Automating and sharing how-to knowledge in the enterprise. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 1719–1728.

[41] Yunyao Li, Rajasekar Krishnamurthy, Shivakumar Vaithyanathan, and H. V. Jagadish. 2006. Getting work done on the web: Supporting transactional queries. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 557–564.

[42] Siân E. Lindley, Sam Meek, Abigail Sellen, and Richard Harper. 2012. It's simply integral to what I do: Enquiries into how the web is weaved into everyday life. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, New York, NY, 1067–1076.

[43] Ronald Mace. 1991. Accessible environments: Toward universal design. In *Design Interventions: Toward a More Humane Architecture,* Wolfgang F. E. Preiser, Jacqueline Vischer, and Edward White (Eds.). Van Nostrand Reinhold, New York, NY, 1–32.

[44] Jalal U. Mahmud, Yevgen Borodin, and I. V. Ramakrishnan. 2007. CSurf: A context-driven non-visual web-browser. In *Proceedings of the 16th International Conference on World Wide Web*. ACM, New York, NY, 31–40.

[45] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Vol. 2. 1003–1011.

[46] Alisha Pradhan, Kanika Mehta, and Leah Findlater. 2018. Accessibility came by accident: Use of voice-controlled intelligent personal assistants by people with disabilities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 459.

[47] Gollapudi V. R. J. Sai Prasad, Sridhar Chimalakonda, Venaktesh Choppella, and Y. Raghu Reddy. 2017. An aspect oriented approach for renarrating web content. In *Proceedings of the 10th Innovations in Software Engineering Conference*. 56–65.

[48] Yury Puzis, Eugene Borodin, Faisal Ahmed, Valentine Melnyk, and I. V. Ramakrishnan. 2011. Guidelines for an accessible web automation interface. In *Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 249–250.

[49] Yury Puzis, Yevgen Borodin, Faisal Ahmed, and I. V. Ramakrishnan. 2012. An intuitive accessible web automation user interface. In *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility*. ACM, New York, NY, 41.

[50] Yury Puzis, Yevgen Borodin, Rami Puzis, and I. V. Ramakrishnan. 2013. Predictive web automation assistant for people with vision impairments. In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, New York, NY, 1031–1040.

[51] I. V. Ramakrishnan, Vikas Ashok, and Syed Masum Billah. 2017. Non-visual web browsing: Beyond web accessibility. In *Proceedings of the International Conference on Universal Access in Human-Computer Interaction*. 322–334.

[52] Readability. 2015. Readability: A Free Web & Mobile App for Reading Comfortably. Retrieved March 18, 2022 from https://web.archive.org/web/20150616195451/https://readability.com/.

[53] Mark A. Rosso. 2005. *Using Genre to Improve Web Search*. Ph.D. Dissertation. University of North Carolina at Chapel Hill.

[54] Mark A. Rosso. 2008. User-based identification of web genres. *Journal of the American Society for Information Science and Technology* 59, 7 (2008), 1053–1072.

[55] Dmitri Roussinov, Kevin Crowston, Mike Nilan, Barbara Kwasnik, Jin Cai, and Xiaoyong Liu. 2001. Genre based navigation on the web. In *Proceedings of the 34th Hawaii International Conference on System Sciences*. IEEE, Los Alamitos, CA, 4013.

[56] Nuzhah Gooda Sahib. 2011. Investigating the information seeking behaviour of blind searchers on the web. In *Proceedings of the 25th BCS Conference on Human-Computer Interaction*. 558–560.

[57] scikit-learn. 2017. scikit-learn: Machine learning in Python. Retrieved March 18, 2022 from http://scikit-learn.org.

[58] Scrapy. 2017. Scrapy: A Fast and Powerful Scraping and Web Crawling Framework. Retrieved March 18, 2022 from https://scrapy.org.

[59] SimilarWeb. 2018. SimilarWeb | Website Traffic Statistics & Market Intelligence. Retrieved March 18, 2022 from https://www.similarweb.com.

[60] Splash. 2017. Splash: A JavaScript Rendering Service. Retrieved March 18, 2022 from https://splash.readthedocs.io.

[61] Abigale J. Stangl, Esha Kothari, Suyog D. Jain, Tom Yeh, Kristen Grauman, and Danna Gurari. 2018. BrowseWithMe: An online clothes shopping assistant for people with visual impairments. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 107–118.

[62] Edward Steinfeld. 1994. The concept of universal design. In *Proceedings of the 6th Ibero-American Conference on Accessibility*.
[63] Molly Follette Story. 1998. Maximizing usability: The principles of universal design. *Assistive Technology* 10, 1 (1998), 4–12.
[64] Markus Strohmaier, Mathias Lux, Michael Granitzer, Peter Scheir, Sotirios Liaskos, and Eric Yu. 2007. How do users express goals on the web? An exploration of intentional structures in web search. In *Proceedings of the 2007 International Conference on Web Information Systems Engineering (WISE'07)*. 67–78.
[65] Yu Su, Ahmed Hassan Awadallah, Madian Khabsa, Patrick Pantel, Michael Gamon, and Mark Encarnacion. 2017. Building natural language interfaces to web APIs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, New York, NY, 177–186.
[66] Zan Sun, Jalal Mahmud, I. V. Ramakrishnan, and Saikat Mukherjee. 2007. Model-directed web transactions under constrained modalities. *ACM Transactions on the Web* 1, 3 (2007), 12.
[67] Hironobu Takagi, Shinya Kawanaka, Masatomo Kobayashi, Takashi Itoh, and Chieko Asakawa. 2008. Social accessibility: Achieving accessibility through collaborative metadata authoring. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 193–200.
[68] Tampermonkey. 2017. Home Page. Retrieved March 18, 2022 from https://tampermonkey.net.
[69] Markel Vigo and Simon Harper. 2013. Coping tactics employed by visually disabled users on the web. *International Journal of Human-Computer Studies* 71, 11 (2013), 1013–1025.
[70] WebAIM. 2013. Survey of Users with Low Vision Results. Retrieved March 18, 2022 from http://webaim.org/projects/lowvisionsurvey/.
[71] WebAIM. 2015. Screen Reader User Survey #6 Results. Retrieved March 18, 2022 from http://webaim.org/projects/screenreadersurvey6/.
[72] Thijs Westerveld, Wessel Kraaij, and Djoerd Hiemstra. 2001. Retrieving web pages using content, links, URLs and anchors. In *Proceedings of the 10th Text Retrieval Conference (TREC'01)*, Vol. 1. 663–672.
[73] William E. Winkler. 1999. *The State of Record Linkage and Current Research Problems*. Statistical Research Division, U.S. Census Bureau, Suitland, MD.
[74] Maayan Zhitomirsky-Geffet, Judit Bar-Ilan, and Mark Levene. 2016. Testing the stability of "wisdom of crowds" judgments of search results over time and their similarity with the search engine rankings. *Aslib Journal of Information Management* 68, 4 (2016), 407–427.
[75] Sven Meyer Zu Eissen and Benno Stein. 2004. Genre classification of web pages. In *Proceedings of the Annual Conference on Artificial Intelligence*. 256–269.

Received February 2020; revised February 2022; accepted February 2022