ASSETS: U: Web Accessibility Evaluation with the Crowd: Rapidly Coding User Testing Video

Mitchell Gordon
University of Rochester
Rochester, NY 14627
m.gordon@rochester.edu

ABSTRACT

User testing is an important part of web accessibility evaluation. However, evaluating the results of user accessibility testing can take a significant amount of time, training, and effort. Some of this work can be offloaded to others through coding video data from user tests to systematically extract meaning from subtle human actions and emotions. However, traditional video coding methods can take a considerable amount of time. We have created Glance, a tool that uses the crowd to allow researchers to rapidly query, sample, and analyze large video datasets for behavioral events that are hard to detect automatically. In this paper, we show how Glance can be used to quickly code video of users with special needs interacting with a website. Then, we show that by coding for whether or not websites conform with accessibility guidelines, we can quickly and easily evaluate how accessible a website is and where potential problems lie.

Categories and Subject Descriptors

K.4.2 [**Social Issues**]: Assistive technologies for persons with disabilities; H.5.m [**Information Interfaces and Presentation**]: Misc.

General Terms

User studies, Experimentation, Human Factors.

Keywords

Accessibility, video, data analysis, crowdsourcing

1. PROBLEM & MOTIVATION

User testing is considered an important part of web accessibility evaluation. In the W3C's Website Accessibility Conformance Evaluation Methodology draft [13], they recommend involving people with disabilities as part of the evaluation methodology. The W3C also published Web Content Accessibility Guidelines (WCAG 2), which help identify the types of important questions to ask when evaluating website accessibility. [2]

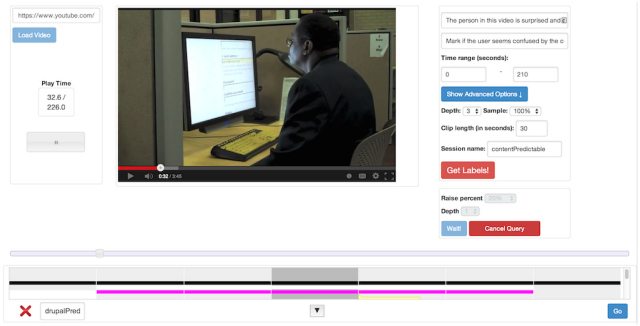

Figure 1: The Glance user interface. Glance can code events in user testing video quickly and accurately. When a usability question is asked, small clips from the video are sent to crowd workers who label events in parallel. The judgements are then quickly merged together and displayed. In this example, we use Glance to determine if and when a disabled user encountered unpredictable content.

However, directly communicating with people with disabilities during the evaluation process may require both significant time and training, which developers frequently lack [10]. While some testing can be performed remotely, the nature of self-reports mean that the data collected may be limited when compared to in-person studies [11].

In addition to user testing, automated testing is also commonly used to evaluate the accessibility of a website [11]. This type of testing is significantly easier to run, quicker, less costly to evaluate than user testing, and often consists of simply entering a website URL and receiving a list of possible accessibility issues. Though not evaluated in the context of accessibility, the crowd-powered system PatFinder is able to use video of users performing tasks to identify higher-level interaction patterns, which describe how to complete tasks such as 'buying a book about HCI.' [7] However, automated techniques have not previously been able to evaluate subtle human actions and emotions that can result from users during user testing [11].

The goal of this work is to combine the effectiveness of user testing with the speed and ease of automated testing. This submission introduces the idea of using Glance [9], a crowd-powered video coding tool, as a way to code video recordings of user testing and significantly decrease

the time and cost associated with evaluating the results of a user test. We validate our claims by running an evaluation using with crowd workers from Amazon Mechanical Turk analyzing videos containing both disabled and non-disabled users performing tasks on websites.

2. BACKGROUND & RELATED WORK

2.1 Video Coding

Behavioral video coding allows researchers in the social sciences to study human interactions [3]. In HCI, researchers often use video coding to discover how users interact with technology [6], and to help better explain those interactions [3].

Video coding is important because it provides a systematic quantitative measure of behavior. However, it can be a very time-consuming task. Researchers often need to code hundreds of hours of video, and the process can take 5-10x longer than the play time of the video itself [1]. Additionally, video coding requires a significant amount of overhead. In order to perform video coding on data, researchers must develop a reliable coding scheme, acquire and train coders, and check for inter-rater reliability. All these factors combined means that performing video coding to evaluate user tests has previously had a very high barrier to entry.

2.2 Glance

Glance is a system that allows researchers to analyze and code events in large video datasets by segmenting videos into short clips and then parallelizing the video coding process across a crowd of online workers (Figure 1) [9]. This approach significantly reduces the amount of time required to gather information from video data, and allows video to be coded in a fraction of the actual play time, depending on the size of the available crowd. Additionally, a large advantage of this speedup is the conversation-like interaction with video data that is enabled by rapid video coding. This type of interaction allows video analysts to more quickly get a sense for their video data and iteratively refine the way that they analyze it in ways that were not previously possible.

To ensure accuracy, Glance can distribute the same video segments to multiple unique workers, and then calculate the variance to provide quick feedback. Glance then uses multiple statistical algorithms to intelligently cluster and combine the segments received from many workers. Glance provides a front-end interface for analysts to enter natural-language queries and to present a visualization of the coding results as they arrive.

This paper extends this prior work by investigating how web developers can leverage the crowd for video coding in order to make their websites more accessible to disabled users.

3. APPROACH

3.1 User Testing Evaluation With The Crowd

Coding video with Glance to evaluate the results from user studies can significantly reduce the amount of time and effort required to obtain actionable information from user tests using the power of the crowd. Additionally, the time, cost, and effort savings that Glance affords makes the use of video coding to evaluate user tests feasible for a larger number of web developers.

Figure 2: The Glance worker interface in "gist" mode. This mode asks workers to simply mark if any instance of the event occurs in the clip, rather than asking them to mark the exact range in which all events occur in the clip. Workers on Amazon Mechanical Turk

We can use the WCAG 2 guidelines, created by the W3C, to determine what to code video of user studies for. According to best practices in video coding, it is important to only ask workers to look for just one event type at a time in order to get an accurate result [1]. Therefore, rather than giving workers a list of things to look for, we must give them a specific type of even to spot. For example, some questions we might ask are "did the user encounter this type of problem: no caption or other alternative provided for multimedia" or "did the user encounter unreadable or difficult to understand text?". By using sighted crowd workers as video coders, we are able to identify problems which may not be identified through remote self-reported accessibility evaluations [11] without requiring developers to take time from the development process to conduct usability testing themselves.

This process works best when using videos of user tests that consist of users attempting to complete some task (on a website, mobile application, etc) and verbally report any issues they run into or concerns they may have. The video should contain a video feed of user's screen, audio containing the user's verbal feedback, and ideally it also contains video of the user them-self (either via picture-in-picture, or via a camera that captures both the user and their screen.

While Glance is able to let workers mark specific segments of time when individual events occur in each clip, we do not believe this is the best approach for this use case. We decided to use Glance's "gist" mode (figure 2), which asks workers to simply mark if any instance of the event occurs

WCAG 2 Guideline	Re-worded instruction
Make content appear and operate in predictable ways	Mark if the content of the website that the user is browsing does not seem to operate in a predictable way
Help users navigate and find content	Mark if the person in the video has trouble navigating the web page or finding content on the web page
Help users avoid and correct mistakes.	Mark if the website did not help the user avoid making a mistake
Give users enough time to read and use content.	Mark if the user did not have enough time to read and use content

Table 1: A summary of the WCAG 2 guidelines that workers looked for during experiment one, and how exactly those guidelines were presented to workers.

within a small clip, rather than asking them to mark the exact range in which it occurs. We believe that gist mode is appropriate for this scenario because determining, for example, exactly what time a user starts and stops being confused is highly subjective and may require additional context that only the web developer is able to provide. Since gist mode operates on only small clips, it is not time-consuming for the developer to determine exactly what causes the confusion.

The end result of this process is that, rather than using a fully-crowd analysis, we assume that the developer will be better at hashing out the details of what exactly is causing issues between the user and their system. But the crowd can still save web developers hundreds of hours of viewing video or directly participating in user tests.

3.2 Envisioned Interaction

Traditionally, the workflow to evaluate the results of user testing has been to either sit with the user while they use the website and note the issues that the encounter, or to painstakingly watch recordings of these tests. Both of these methods take significant time because they require the web developer to be completely engaged with the test for the its full length, which can often take hours per user.

This project makes evaluating user tests significantly less time-consuming for web developers because they do not need to be concerned with fully participating in or viewing the user test them-self. Rather, the web developer will only need to view the specific portions of the user test that uncovered an issue. We envision the interaction to be:

1. Obtain recorded videos of user tests for the website.
2. Determine what WCAG 2 standards might apply to the website, and use the Glance interface to easily select which of them to test for.
3. Wait while crowd workers use Glance to perform the evaluation for you. Our prior work has found that Glance can code nearly 50 minutes of video in 5 minutes [8]. To recruit workers, we recommend using LegionTools [5].
4. View the Glance results page to see what parts of the user test that workers found WCAG issues occurred. (Figure 1).
5. Review the clips that the crowd found to see what the specific issue was, and fix it on the website to improve accessibility.

4. EVALUATION AND RESULTS

To evaluate Glance's ability to code video of accessibility user testing and return reliable results, we ran a feasibility experiment using Glance's "gist" mode.

For pragmatic reasons, our experiment consists of two parts: one which contains non-disabled users and another that contains disabled users. We followed the trial setup used by a W4A paper by Puzis et al [12]. In their evaluation, they first used non-disabled users to verify that their system worked at all. This was a task that could be performed equally well using either non-disabled or disabled users, but they used non-disabled because doing so was faster and easier. They then used disabled users for a further evaluation task that could only be performed with disabled users.

We followed this same study setup for similar pragmatic reasons as Puzis et al. There are a wide range of user testing videos involving non-disabled users available on the web, but a much smaller number of user testing videos that involve disabled users are available. Producing our own user-testing videos involving disabled users for a larger evaluation is ongoing work (the process for doing so is extremely high-effort and time consuming).

First, we do a larger study with user test videos that contain non-disabled users. This allows us to verify that Glance can be used to accurately code typical user testing videos containing a wide variety of participants and actions. This is equally valid to perform with non-disabled users as with disabled users because while the problems that non-disabled users hit are not necessarily the same as those that disabled users hit, both groups typically need to accomplish the same tasks (e.g. sign up for an account, buy something off Amazon, etc) using the same websites. Both groups will still run into usability issues that violate WCAG 2, but the specific guidelines that come up may be different.

We follow up with smaller study that focuses on disabled users, in order to show that workers are still able to see what it looks like when disabled users hit problems.

We recruited all workers on Amazon Mechanical Turk using LegionTools [5] and paid each worker 15 US cents per task. Each task took workers 60 seconds to complete, making the effective hourly wage $9/hr.

4.1 Typical User Tests

We ran Glance on five video clips, each containing one participant, making five unique participants total. Four of those videos contained a single WCAG 2 guideline violation while one video did not contain any. To test against false-positives, each video was run for all the guidelines that we tested for. Each video was coded by at least 3 unique

workers, for a total of 60 workers. The videos all displayed non-disabled users completing simple tasks on websites and talking out loud about their thought process. Each video contained a video feed of the user's screen, the user themself, and the verbal audio feed from the user. When asking workers what to look for, we slightly modified the WCAG guidelines to make workers tasks clearer. Table 1 shows the specific guidelines and text that we asked workers about.

This study returned results of 93.3% precision (the number of times that workers correctly said an event occurred out of the total number of times workers said an event occurred) and 87.5% recall (the number of times that workers correctly said an event occurred out of the number of times workers should have found an event).

4.2 User Tests Involving Disabled Users

This evaluation used a video of a user test that consisted of three visually-impaired participants completing tasks on a website using a desktop computer, and providing verbal feedback on any issues they were running into. We coded for the WCAG 2 guideline "make content appear and operate in predictable ways". This small evaluation consisted of 21 Mechanical Turk workers coding a three and a half minute video split into 7 30-second segments. Each segment was redundantly coded by three unique workers.

The crowd correctly coded clips with a precision of 88.3% and recall of 99.3%.

4.3 Discussion

These results show that the crowd is able to accurately determine whether a violation of specific WCAG 2 guidelines occurs in videos involving both typical users and involving disabled users. These accuracy levels are in line with the initial verification of the Glance system in its UIST paper, showing that showing that video coding of user tests is no less accurate than other anticipated uses of Glance and video coding in general. Our tests with video containing typical users shows that the crowd was accurate for a wide variety of WCAG 2 guidelines, and our test with video containing disabled users showed that the crowd did not become noticeably less accurate when viewing video of disabled users. This shows that the Glance system can accurately detect WCAG 2 issues in user tests of websites involving disabled users.

4.4 Limitations

While the ideas presented in this paper make the process of conducting user testing with disabled users significantly faster and easier than conventional methods, there are currently a few limitations.

The first limitation lies with our use of the WCAG 2 guidelines for accessible web content. These guidelines are very specific, and since with video coding it is necessary to code just for one feature at a time, it is difficult to code for a large number of small features. To combat this, we gave more general instructions based off of a summary of the guidelines posted by the W3C called "WCAG 2 at a Glance." However, this may result in slightly less precise results than if we were specifically coding for every feature mentioned in each WCAG 2 guideline. We present a possible solution to this limitation in the Future Work section of this paper.

A second limitation lies with the Glance system: lack of context. Currently, Glance routes random video segments, typically between 30 and 60 seconds in length, to random workers. If some feature requires context from multiple video segments to identify, it may be difficult for workers to detect. The Glance UIST paper addresses this issue, and we do not believe that significant context is necessary to detect the majority of WCAG 2 guidelines. We did not run into situations where context appeared to be an issue in the evaluation for this paper.

5. FUTURE WORK

We would like to expand our evaluation to include a larger variety of user testing videos and code them for other sets of accessibility guidelines besides just the WCAG 2 guidelines. We currently focus on just the WCAG 2 guidelines when deriving what features to code video for. In the future, we may see even better results if we map these guidelines to lower level features that are more similar to the commonly used Specific Affect Coding System (SPAFF) [4] and other more traditional coding tasks.

As mentioned in the evaluation section of this paper, we are currently working on creating a new set of data involving user testing video from a large number of disabled users. In doing this, we will also be able to test the process of collecting user testing video data from disabled users.

We also believe that, in addition to website user test evaluations, Glance has many possible applications within accessibility. These are limited only by what types of videos can be created and by what types of events can be accurately coded for. Some of these possibilities include:

- Visually-impaired users taking a panoramic video with their phone and asking a question about it.
- Disabled users can upload a video of their interaction with a website or app that is not accessible. If they were not able to completed a desired action in the inaccessible app, they could ask a question to help figure out where in the process of using the app they went wrong.

6. CONCLUSION

We have presented a significantly easier and faster way for developers to evaluate the results of user testing involving disabled users. We use our Glance system, which uses the power of the crowd to quickly and accurately code video, to determine when accessibility issues arise in user tests. This improves the user testing process because it takes away the pain of evaluating the results of the user test, which previously took a great deal of time from web developers to do.

We believe that by making the user testing process easier and faster, more web developers will be willing to perform user tests. User testing is critical in making websites more accessible to disabled users, so this is an important step towards making websites accessible to everyone.

7. REFERENCES

[1] *Handbook of Research Methods in Social and Personality Psychology.* Cambridge University Press, 2000.

[2] Web content accessibility guidelines (wcag) 2.0. web, Dec. 2008.

[3] R. Bakeman and J. M. G. PhD. *Observing Interaction: An Introduction to Sequential Analysis*. Cambridge University Press, 1997.

[4] J. M. Gottman, K. McCoy, J. A. Coan, and H. Collier. The specific affect coding system (spaff) for observing emotional communication in marital and family interaction. 1995.

[5] https://github.com/RocHCI/LegionTools. *LegionTools: an easy way to recruit and route workers from Amazon Mechanical Turk.*

[6] B. Jordan and A. Henderson. Interaction analysis: Foundations and practice. *The Journal of the Learning Sciences*, 4(1):pp. 39–103, 1995.

[7] W. Lasecki, T. Lau, G. He, and J. Bigham. Crowd-based recognition of web interaction patterns. In *Adjunct Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST Adjunct Proceedings '12, pages 99–100, New York, NY, USA, 2012. ACM.

[8] W. S. Lasecki, M. Gordon, S. P. Dow, and J. P. Bigham. Glance: Enabling rapid interactions with data using the crowd. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, pages 511–514, New York, NY, USA, 2014. ACM.

[9] W. S. Lasecki, M. Gordon, D. Koutra, M. Jung, S. P. Dow, and J. P. Bigham. Glance: Rapidly coding behavioral video with the crowd. UIST 2014, New York, NY, USA, 2014. ACM.

[10] J. Lazar, A. Dudley-Sponaugle, and K.-D. Greenidge. Improving web accessibility: a study of webmaster perceptions. *Computers in Human Behavior*, 20(2):269–288, 2004.

[11] J. Mankoff, H. Fait, and T. Tran. Is your web page accessible?: A comparative study of methods for assessing web page accessibility for the blind. CHI '05, pages 41–50, New York, NY, USA, 2005. ACM.

[12] Y. Puzis, Y. Borodin, R. Puzis, and I. Ramakrishnan. Predictive web automation assistant for people with vision impairments. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 1031–1040, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

[13] W3C. *Website Accessibility Conformance Evaluation Methodology (Working Draft)*, 2014 (accessed June 10, 2014).