FourEyes: An analysis of user interfaces for large-scale crowdsourcing of visual attention data

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

In this work, we compare and analyze four UIs for collecting crowdsourced attention maps. We introduce ZoomMaps, which allows collecting attention data at multiple image scales using the familiar zoom gesture on a mobile phone. We conduct the first in-depth analysis of CodeCharts, an interface for approximating eye locations at precise viewing durations. These novel interfaces are compared to ImportAnnots, a tool for collecting annotations of graphic design importance, and the moving-window interface BubbleView. We lay out in detail the methodology used for each of these interfaces and discuss which types of images and potential applications are appropriate for each. Additionally, we collect CodeCharts, ZoomMaps, and ImportAnnots data on five types of image stimuli–natural images, resumes, graphic designs, infographics, and academic posters–and analyze the results to characterize in detail what can be learned from each method of data collection.

ACM Classification Keywords

H.5.1. Information Interfaces and Presentation: Miscellaneous

Author Keywords

Eye tracking; attention; crowdsourcing; interaction techniques

INTRODUCTION

The eye-catching parts of a photograph or infographic will be the ones for which the image will be later remembered [3]. The most important parts of a design can be used to summarize the design for later retrieval [5]. The most salient parts of images can guide automatic image cropping and retargeting [7]. All these applications require an understanding of where people look in images and designs.

CHI'16, May 07–12, 2016, San Jose, CA, USA

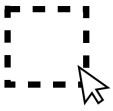
ISBN 123-4567-24-567/08/06…$15.00

DOI: http://dx.doi.org/10.475/123_4

Figure 1. In this paper we consider different approaches for crowdsourcing human visual attention patterns without the use of an eye tracker. This figure features two of the user interfaces from this paper. As shown in the top two rows, the attention maps generated by CodeCharts and ZoomMaps can mimic heatmaps obtained using direct eye tracking. The last row shows where ZoomMaps data starts to diverge from the other methodologies, as observers explore more distant content. This paper will cover the similarities between the methodologies in capturing stable aspects of human attention, as well as the unique features that make the interfaces suitable for different applications.

ZoomMaps: participants use the pinch-zoom gesture on their mobile phone to explore image content at higher resolution.

CodeCharts: participants self-report a region of an image they gazed at using a grid of codes that appears after image presentation.



ImportAnnots: participants are asked to paint over regions of a design that they consider important using binary masks.

BubbleView: participants click to deblur/expose small bubble regions on an otherwise blurry image.

Decades of work investigating human attention have depended on eye movement data collected using eye tracking hardware. Computational models built for predicting human attention on images for applications like cropping and retargeting have in turn been trained and evaluated on datasets of eye movements.

Data collection using an eye tracker is hard to scale given that each recording session requires an experimenter to be present and participants to come into the lab. This also makes iterating on experimental design choices time-consuming, requiring recruiting a fresh batch of participants to come into the lab for every experimental variation.

Motivated by these shortcomings of the standard procedure of collecting human attention data via eye-tracking, we look towards methods to approximate eye movements using web-based experiments. In this paper, we analyze four user interfaces that are deployable as Amazon's Mechanical Turk (MTurk) tasks for large-scale crowdsourcing of visual attention data (Fig. 1). All the interfaces capture some common aspects of human attention, but are best suited for different image types and tasks. We compare these four interfaces and discuss use cases for each.

We will focus mostly on ZoomMaps and CodeCharts in this paper, as they have not been discussed in prior literature. ZoomMaps is presented for the first time in this paper, while CodeCharts was used for multi-duration data collection in [8] but not carefully analyzed or applied to other use cases. ImportAnnots refers to the interface for capturing explicit "Importance Annotations" that was first introduced by O'Donovan et al.[19]. For the purposes of this paper, we used the annotation tool in our crowdsourcing tasks and set up a validation procedure. BubbleView was the subject of a thorough investigation in [14], so we will use it here as a comparison point.

In summary, our contributions are (1) introducing the ZoomMaps interface, (2) providing the first analysis of the CodeCharts interface, (3) comparing ZoomMaps and CodeCharts to previously presented interfaces, BubbleView and ImportAnnots, (4) providing an analysis and discussion of crowdsourced visual attention data collected using all four interfaces, including the distinct aspects captured by each interface and the different use cases these interfaces are suitable for. This is not a survey paper of user interfaces; rather, this is a deep dive into a set of interfaces that we ran experiments on[1].

RELATED WORK

Previous work has investigated the correlation between cursor and gaze locations [22, 10, 11]. Cursor movements can complement eye movements, especially when a participant can use both to interact with visual content. However, can cursor movements replace eye movements entirely when those are unavailable? A separate line of work has investigated cursor-based interfaces as a proxy for eye tracking [1, 13, 25]. For instance, the moving-window methodology reveals only portions of an otherwise-obscured image depending on where a user positions their mouse cursor [12, 18, 21, 26]. These approaches have inspired the BubbleView methodology [14]. In this paper, we have in turn been inspired by limitations in the BubbleView methodology, which led us to design the ZoomMaps and CodeCharts interfaces. Unlike BubbleView and other moving-window methodologies, ZoomMaps and CodeCharts do not obscure the underlying image content. To scale up attention data collection beyond what is possible with in-lab eye trackers, other researchers have turned to using built-in webcams as a way of getting coarse-grained attention data from crowdworkers [20, 17]. These approaches still suffer from requiring a lot of constraints: particular lighting conditions, stability of the observer in front of camera, etc. None of the user interfaces we consider explicitly measure eye movements or require an observer's eyes to be recorded. Rather, we make use of interaction methodologies that are correlated with visual attention.

[1]The user interfaces presented in this paper will be shared with the research community upon publication.

USER INTERFACES FOR CROWDSOURCING ATTENTION

In this section we present the four interfaces that we will be comparing in the rest of the paper. Because different interfaces are better suited to different tasks (discussed at length later in this work), we do not exhaustively compare all interfaces on all the same images and tasks. For our studies, we collect crowdsourced attention data on natural images from the OSIE [28] and CAT2000 [2] datasets. The latter dataset already had BubbleView data [14], and we additionally collected data at 6 different viewing durations using CodeCharts. For 35 images from the CAT2000 dataset, we have data from CodeCharts, ZoomMaps, and ImportAnnots. Moving away from natural images, we also have attention data on resumes from all three interfaces and data on graphic designs from CodeCharts and ImportAnnots. We downloaded vector templates of 116 resumes and 20 graphic designs from Canva.com. Since on mobile participants have the ability to scale image content, we additionally ran ZoomMaps with larger scaled images than the other interfaces - specifically, infographics (from MASSVIS [3]) and academic posters (from personal collections). To measure similarity of the data collected using the different user interfaces to each other and to eye movements, where appropriate, we use the NSS (Normalized Scanpath Saliency) and CC (Correlation Coefficient) metrics used by [14] and described in detail in [4].

ZoomMaps

The mobile screen provides a naturally restricted window that is frequently used to explore multi-scale content with the help of the zoom functionality. We show that the zoom patterns generated by participants viewing images on their mobile screens can be used as an approximation for visual attention.

Implementation. To capture mobile zoom information, we built an image gallery website with tracking capabilities. The gallery was based on the PhotoSwipe JavaScript library[2], modified to capture and store any changes to the visible region of the image along with a timestamp. The interface allows pinching to zoom in or out on an image and swiping to switch between images (Fig. 2). When the image is swiped, all the interaction data on the image is stored to a sever. The interaction data contains the coordinates of the zoomed-in portion of the image with a timestamp for every *event* triggered by the user (i.e., the image is re-scaled or re-positioned on the screen).

[2]https://github.com/dimsemenov/photoswipe

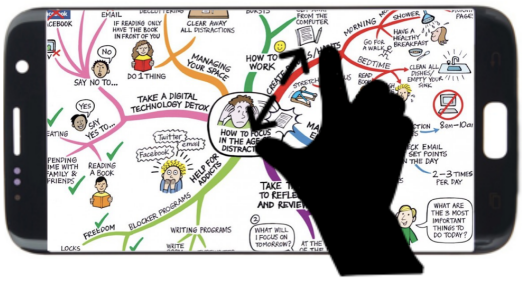


Figure 2. ZoomMaps UI. Participants use the pinch-zoom gesture on their phone to explore image content at higher resolution.

Task flow. When participants open the task, they are given a QR code to open the image gallery on their mobile browser (the URL is shown as well in case they are already on mobile). Upon completion, participants receive a completion code which should be entered on the first page to submit the task. Depending on the experiment, we either insert questions between images (in which case questions need to be answered on mobile) or add a task-specific questionnaire at the end of the task (which can be completed on desktop).

Validation procedure. Once participants reach the final image in the sequence, if they have not spent sufficient time on the task or performed a minimal number of pinch-zoom gestures, the image sequence restarts. Specifically, we require that participants spend at least 1-5 seconds (depending on the experiment) on 85% of the images (even though we instruct participants to spend 5-15 seconds per image) and at least 3 minutes in total on a task that we estimate should take 4-9 minutes. Further, we require that participants zoom on at least 20% of the images. These thresholds were chosen empirically based on pilots. In most image collections, there are at least a few images that are uninteresting, receiving little viewing time, and there are frequently images that require no zooming because all elements are visible at the original scale.

Generating attention maps. Our mobile interface stores the bounding boxes of zoomed image regions within the coordinate frame of the whole image, along with timestamps of when regions were in focus. We use this information to extract which parts of the image were viewed for how long and at what zoom level.

We construct the attention map as follows: for every pixel in the image, we compute its average *zoom level* over the entire viewing interval. We define the zoom level for an image region as the quotient of the full image area divided by the area of the image region that has been magnified. We assign this zoom level to all the pixels contained in the image region. By aggregating these zoom levels over time, we compute each pixel's average zoom level. We can then visualize this value per pixel to obtain a ZoomMaps attention map (Fig. 3). Higher values in attention map correspond to regions of the image that were inspected with closer zoom on average.

Cost. Participants were paid between $1.00 and $1.25 to look at 5-35 images. Payment depended on the type of images workers were asked to examine, how much time they were asked to spend per image, and if they answered questions after every image or just at the end of the task. For twenty impressions per image, data collection cost $0.72 per natural image, $1.33 per resume, and $5 per infographic.

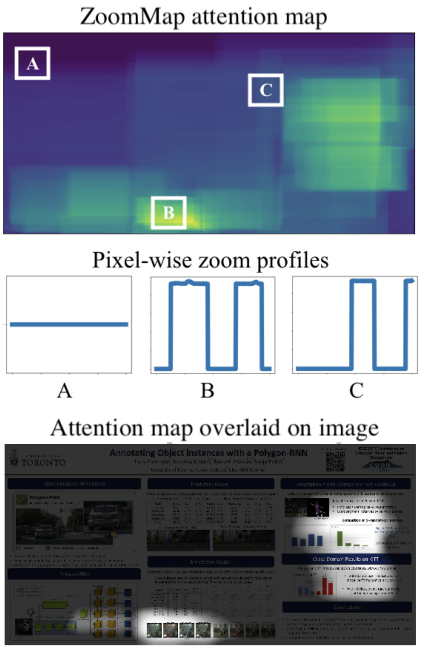


Figure 3. ZoomMaps attention map, visualized alone, at selected pixels, and on top of the image used to create it. For three pixels, labeled A, B, C, zoom over time is averaged to produce a value in the corresponding attention map.

CodeCharts F37

CodeCharts allows for collecting gaze locations at precise viewing durations in a crowdsourced manner by asking participants to self-report where they were looking on an image using a grid of three-character codes called a codechart.

Task flow. After an image is shown on a participant's screen, there appears in the same location a jittered grid of three-character codes. The participant reports the last triplet they saw when the image vanished, which approximates where on the image they were looking at the time. This process, visualized in Figure 4, repeats for several images. Trials are separated by a fixation cross to center the participant's gaze.

Implementation. We pad and resize all input images to a consistent size and generate codecharts with the same dimensions. Three-character codes are placed on the codechart in a grid with a slight random horizontal and vertical jitter. Each time an image is presented it is accompanied by a different code chart with a different jitter which leads to a more even distribution of code placement. When shown in the browser, both target images and code charts are resized to fit into a display window of 1000 by 700 pixels and triplets are displayed at a font size of around 16 pixels.

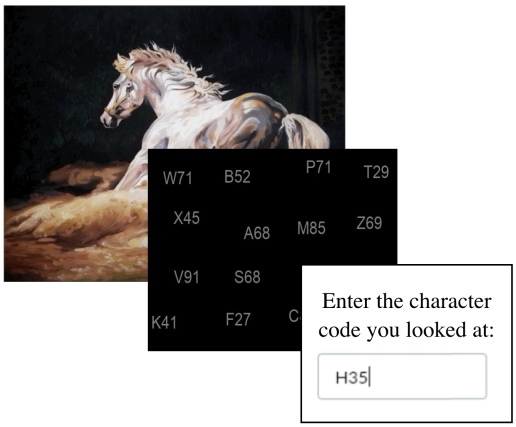


Figure 4. CodeCharts UI. Participants self-report a region of an image they gazed at using a grid of codes that appears after image presentation.

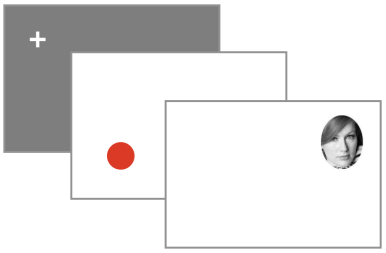


Figure 5. Sample validation images for the CodeCharts UI: we used a regular fixation cross and a salient red circle on a white background, before selecting a cropped face on a white background as the type of validation image that encouraged participants to move their eyes most to the location of the validation cue. Participants were expected to type a triplet code from the codechart immediately following the image that overlapped with the cue. The cues are plotted larger than actual scale for ease of viewing.

We wanted to display the codechart long enough that workers had time to read a triplet but not so long that their eyes could wander. We found that 400ms was the shortest duration that did not see a significant degradation in validation accuracy. By decreasing codechart exposure from 750ms to 400ms, we increased the NSS similarity of CodeCharts data to ground truth eye movements from 1.74 to 1.89.

One limitation of the interface is that gaze locations are effectively quantized by the codechart, which can produce gridlike artifacts in the output. This can be mitigated in the future by jittering the whole grid of codes instead of jittering individual codes. Experimeters must take care that triplets are appropriately spaced and extend to the edge of the image in order to avoid collecting skewed data.

Validation procedure. Validation images consist of a plain background with a single point of interest that aligns with one or more "correct" triplets. We tried three different formats for validation images: a plain fixation cross, a solid red circle on a white background, and a cropped face image on a white background, in that order (see Fig. 5. We found that faces provide a more interesting cue than simpler stimuli, encouraging participants to attend to the cue. We also found that explicitly instructing participants to look at vigilance images increased performance.

Our task starts with a screening phase of three normal and three validation images where workers must correctly enter all validation codes and can only enter one code that is not present on a codechart. Validation images are also interspersed throughout the sequence and are used to discard data from workers who miss over 25% of validation codes. We also eliminate subjects who look at the same spot on the screen for many images in a row. We find that indeed, participants with higher validation accuracy produce data that is more similar to human eye movements, perhaps because these workers are more attentive or get used to moving their eyes in response to stimuli.

Generating attention maps. We blur the gaze points with a sigma of 50 to generate a heatmap. Our triplets are spaced approximately 100 pixels apart, so 50 is a good approximation of the radius of uncertainty in the interface.

Number of participants. Since CodeCharts yields only a single gaze location per subject per image, it requires 50 subjects to get stable results (see Fig. 12).

Cost. Participants spent on average 1 minute reading instructions. For the 48-trial experiment on natural images, they spent just over 6 minutes on the rest of the task including a demographic survey. Paying an hourly rate of $10, this worked out to be $1.25-$2.00 per image for 50 participants.

Likeability. The task was generally very well-received. Crowdsourced workers often described it as "fun" and "interesting" while also being "hard" and "fast". We hypothesize that the automatic timing of the task contributed to a game-like experience.

ImportAnnots

The idea of having crowdworkers annotate important elements on graphic designs using binary masks and then averaging those binary masks to construct heatmaps of image importance was introduced by O'Donovan et al. [19]. Importance maps were collected for over a thousand images and used to train computational models to predict importance on new designs [19, 5]. For this paper, we re-purposed the initial interface, added a validation procedure, and tested the interface on different image types including natural scenes, infographics, and resumes.

Task flow. Participants are presented with images, one at a time, and are asked to annotate the most important regions (Fig. 6). There are no definitions of what should be considered "important". Participants have a maximum of one minute per image.

Implementation. We used legacy code from O'Donovan et al. [19] for the annotation tool embedded within our task interface. It is a Flash application that provides 3 annotation tools: a stroke fill that allows tracing the contours of an object to

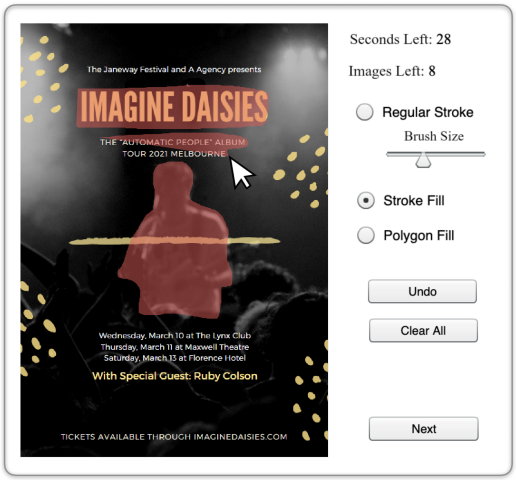


Figure 6. ImportAnnots UI. Participants paint over regions of a design that they consider important using binary masks.

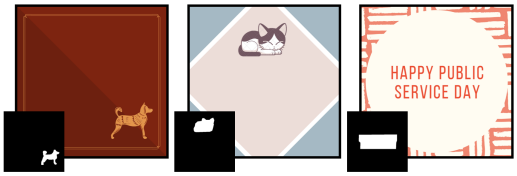


Figure 7. Sample validation images for the ImportAnnots UI: we simplified vector graphic designs to contain a single text or graphical element that unambiguously stood out. Throughout the task, we computed intersection-over-union of crowdworker annotations with ground truth annotations (insets) to ensure annotation quality.

provide a fine-grained segmentation, a polygon fill that allows plotting points with connected lines for coarser annotation, and a regular stroke for painting over a region. Stroke fill is set as the default and that is what most participants choose to use: we observe that 64% of assignments are completed using stroke fill, while polygon fill is used 34.2% of the time and regular stroke only 1.8%. It is interesting to note, however, that users utilizing the polygon stroke tool generate more individual annotations per assignment. They produce 31.8 annotations per assignment on average, while stroke fill users produce 21.1 and regular stroke users generate 23.3.

Validation procedure. Interspersed throughout the task were validation images in the form of graphic designs containing one main text or graphical element (Fig. 7). These validation images were constructed by sampling vector designs and paring them down to make annotation of importance more objective. Forty such designs were created in total, and are randomly sampled for individual tasks. For a 5-minute task, participants annotated 10 images and 3 validation images in a random order. To meet our quality thresholds, participants needed to annotate at least one object in all but one of the images and to correctly annotate 2/3 of the validation images. Correctness was measured by an intersection-over-union (IoU) threshold of 0.55, which was selected by calculating mean IoU across hand-selected good tasks.

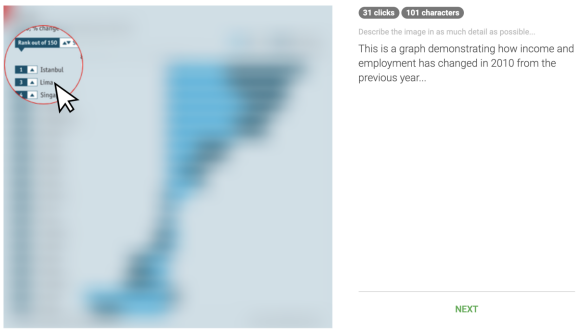


Figure 8. BubbleView UI. Participants click to deblur/expose small regions of an otherwise blurry image.

Generating attention maps. Each participant generates one binary mask per image. The binary maps are average across participants to produce an overall map for the image. It was pointed out in the appendix of [19] that despite high inter-observer annotation variability and noisy annotations, averaged over a large number of users (20-30), the mean importance maps give a plausible ranking of importance.

Cost. We paid participants $0.85-$1.00 for annotating 10 designs, bringing total costs to $2.55-$3.00 per image for 30 annotations.

Likeability. The stroke fill tool is not immediately intuitive. However, it is more efficient than polygonal annotation tools [24]. Crowdworkers picked up on this *"Could be done faster if you are good with the tools"*.

BubbleView

BubbleView is a cursor-based, moving-window methodology. This means that a viewing window is triggered by the participant moving the mouse cursor. Specifically, the original image is blurred to distort any text regions and disable legibility, requiring participants to click around the image to de-blur small, circular "bubble" regions at full resolution (Fig. 8). This was designed to mimic a blurred periphery and the confined area of focus of the human eye fovea [14]. BubbleView was initially introduced in [15] (inspired by prior work [6, 9, 12]) and is tightly related to the SALICON methodology [13], which replaces mouse clicks with continuous mouse movements, and a fixed image blur with a graded one, that is recomputed at each mouse position. In Kim et al. [14], it was shown that BubbleView can more closely approximate ground truth eye fixations. In the latter paper there is a thorough comparison of BubbleView to eye movements on different image types and tasks. We use some of that analysis in the present paper.

Task flow. Given a sequence of images, participants explore one image at a time using their mouse cursor to click and deblur regions of an image (clicking on a new location reblurs the previously clicked location). For free-viewing tasks (no instructions other than to freely explore the image), the viewing time is fixed; while for description tasks, participants are presented with a text box on the side of the image where

they are asked to describe the image. As noted in Kim et al. [14], it takes 2-3 times longer for participants to click to explore an image than to freely view it (to generate the same number of gaze points on an image in the same unit time). In other words, this interfaces slows down visual processing relative to natural viewing.

Implementation. BubbleView exposes several parameters to the experimenter: the blur sigma used to distort the image, the radius of the bubble region exposed on-click, the task timing and set-up (whether or not to include a required text field entry with each image), and whether discrete mouse clicks or continuous mouse movements (like SALICON [13]) are to be collected. Different task types warrant different parameter choices to best approximate ground truth eye movements. We refer to Kim et al. [14] for the details.

Validation procedure. At task-time the only validation procedure was a minimum number of 150 characters required in the text entry box for describing each image. During post-processing, participant data that included too few clicks was removed. These thresholds were set empirically (to 10 clicks/image for a description task and 2 clicks/image for a free-viewing task [14]). Interquartile range-based outlier computation [16] was also used to remove too few or too many clicks as a secondary threshold.

Generating attention maps. Given a set of BubbleView mouse clicks on an image, an attention map is computed by blurring the click locations with a Gaussian with a particular sigma (a different one per image dataset [14]).

Cost. Adjusting BubbleView hourly rate from $6 to $10 (to be comparable to the compensation we use in our tasks below), we can re-estimate the numbers in Table 7 of [14]: $0.44 per image for 15 participants and a free-viewing duration of 10 seconds.

Likeability. Crowdworkers' experience with this interface depended in part on the experimental parameters used. Kim et al. [14] indicated that participants complained about the difficulty and tediousness of the task at small bubble sizes.

DIFFERENT UIS FOR DIFFERENT APPLICATIONS

Figure 9. Can you guess which attention map was generated with which UI? Answers at the end of this section. Level sets visualized to improve discriminability.

Type of stimuli

Some stimuli are better-suited to particular interfaces. One important consideration is scale of the stimuli. ZoomMaps, ImportAnnots, and BubbleView allow panning/scrolling and therefore are compatible with images larger than the screen dimensions. By contrast, CodeCharts' automatic task progression requires that the stimuli fit on the screen at once so that the participant can take in the image at a glance and decide where to look.

Furthermore, only one interface –ZoomMaps– supports viewing images at varying resolutions. This makes it uniquely qualified to collect detailed viewing data on images that have content at multiple scales, such as academic posters, infographics, or crowded pictures.

ImportAnnots is most appropriate for easily-segmentable, non-natural images.

Although not explored in this paper, CodeCharts can also be used to collect gaze data on video, as suggested in [23].

As shown in Fig. 9, the attention maps collected can vary dramatically across interfaces for certain images. The maps visualized in order are: CodeCharts, ImportAnnots, and ZoomMaps. The CodeCharts attention map focuses on faces, much as human eye movements would; the ImportAnnots map highlights the full object that is the main focus of the photograph; ZoomMaps attention map includes points of focus in the distance of the image, where there are other small people on the mountainside.

Type of task

Some of the interfaces lend themselves to particular tasks. Table 1 provides a high-level overview of the tradeoffs of each interface.

ImportAnnots produces high-fidelity element segmentations, making it ideal for measuring graphic design importance but awkward for natural images.

CodeCharts is the only interface where stimuli exposure is carefully controlled by the experimenter. Gaze points can be collected at precise viewing durations without distorting the underlying image or impeding the natural viewing process like in BubbleView. This capability was used in [8] to collect attention data at various viewing durations for the same image. The downside is that CodeCharts only allows for collecting one data point per image per participant, so intra-participant analysis is not possible.

ZoomMaps allows for collecting attention data on multi-scale images via a familiar interface, but the mobile screen provides relatively coarse-grained data.

BubbleView is perhaps the most versatile interface, working for natural and non-natural images at a variety of sizes. It is also the cheapest. However, it distorts the underlying image and slows down the viewing process, which can introduce artifacts.

TAKE-AWAYS

We use data pulled from each of our main categories of stimuli and collected using all four interfaces to better understand the applications and capabilities of each of the UIs.

Different data density requires different numbers of participants.

We use a similar analysis as in Kim et al. [14] to determine how many participants we need to obtain stable results for

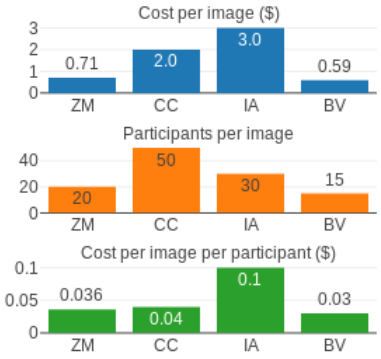


Figure 10. Cost and number of participants comparison across the different interfaces.

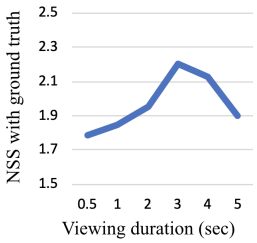


Figure 11. Plot showing the NSS score of CodeChart data at varying durations measured against OSIE eye-tracking data. The peak is at 3 seconds, which notably is the duration for which OSIE data was collected.

each methodology (Fig. 12). It was reported that after about 10-15 participants, the similarity of BubbleView clicks to ground truth fixations was already 97-98% of the performance achievable in the limit. We collect many participants' worth of data for a pilot study for each of our user interfaces and use the performance with the maximum number of participants as an approximate limit. We find that CodeCharts, because it only collects one gaze location per worker per image, requires 50 workers per image, significantly more than BubbleView. ZoomMaps requires 15-20 participants to achieve 98% of the performance with the maximum number of participants. For ImportAnnots, because direct comparison to eye movements is not as meaningful, we consider the point where the ranks of the design elements obtained by averaging the annotations of a set number of participants is closely aligned (Spearman's rank correlation = 0.98) with the ranks obtained by using all participants. This happens at 30 participants.

BubbleView and CodeCharts are most similar to human eye movements.

We ran data collection using CodeCharts, ZoomMaps and ImportAnnots on a set of 35 images sampled from the CAT2000 dataset [2]. CodeCharts achieved an average NSS score of 2.00 at predicting ground truth eye movements; ZoomMaps achieved an NSS score of 1.37, and ImportAnnots a score of 1.22. Fig. 19 includes some representative examples. Across 31/34 images, CodeCharts best approximated eye movements, focusing on a few key regions on an image, particularly faces. On 23/35 images, ZoomMaps attention maps more similar to eye movements than ImportAnnots. The middle row of Fig. 19 shows a sample case where ZoomMaps has an over-focus on

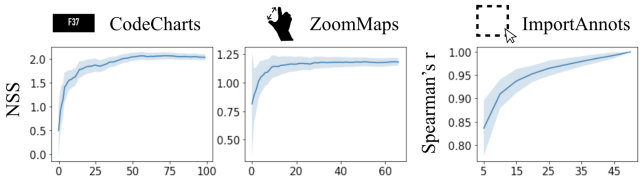


Figure 12. Plot of how performance improves with number of participants for CodeCharts, ZoomMaps, and ImportAnnots. We choose our recommended number of participants based on the number that acheives 98% of the accuracy of the full number of participants.

distant elements at the expense of highlighting foreground objects. ImportAnnots tends to highlight a few objects per scene, ascribing uniform importance to most object regions. However, human gaze (whether collected using an eye tracker or using the CodeCharts UI) falls on certain object regions only (e.g., faces, hands, points of contact, etc.).

Given that CodeCharts best approximated eye movements, we further collected CodeCharts attention data on the same subset of the OSIE dataset [28] that both BubbleView and SALICON [13] were evaluated on in [14]. We collected CodeCharts data for 51 participants at each of 6 image viewing durations: 0.5, 1, 2, 3, 4, and 5 seconds. We found that CodeCharts best matched OSIE's ground truth eye movements at a viewing duration of 3 seconds (Fig. 11), which is, notably, the duration that was used for the eye tracking experiments [28]. At this duration, we obtained an NSS score of 2.20 (compared to BubbleView at 2.69 and SALICON at 2.61). While CodeCharts achieves a lower NSS score, it is better suited for approximating what participants might see at a given viewing duration and across durations. For the OSIE data, BubbleView data was collected with 10 seconds of viewing per image, and SALICON data was collected at 5 seconds, both chosen somewhat arbitrarily to compensate for the extra time it takes participants to click or move the mouse cursor, respectively.

Figure 13. Natural-image examples where CodeCharts and ImportAnnots do and do not agree. Top row: The interfaces give differing predictions. The CodeCharts attention map mirrors the center bias of eye movements and reflects explorations of the image, whereas ImportAnnots finds clear objects of interest in the image and attempts to segment them. Bottom row: The two interfaces are well-aligned because the animals are both salient and easily-segmented.

Attention on an image is worth two cents (+/- 1)

Across interfaces, there seems to be a remarkable consistency in cost per image trial. Dividing the total task cost by number of participants, we calculated that we paid on average 2-4 cents per trial per participant in the CodeChart experiments, where

Figure 14. Examples of CodeCharts and ImportAnnots performance on graphic designs. CodeCharts data highlights the center of the image while ImportAnnots focuses on text regardless of position. Top row: CodeCharts and ImportAnnots are well-aligned because the title of the graphic is centered in the design. Middle row: The title is not salient but is still the most important element. Bottom row: An example where a pictoral element is the most important. Notice how saliency is concentrated at a particular point (the cat's face) whereas importance segments out the entire photo element.

the variance depends on image presentation duration (ranging from 0.5 seconds to 6 seconds). The ZoomMaps price of $1.25 per 35-trial task translates to 3.6 cents per image. Table 7 of [14] can be used to calculate that the per-image cost of BubbleView is roughly 3 cents for a viewing duration of 10 seconds. As a comparison point, for in-lab eye-tracking participants, our hourly rate is around $20. In a single one-hour sitting, participants can go through about 1000 images, with a typical viewing duration of 2-3 seconds each. This works out to be 2 cents per image. Unlike the other interfaces, ImportAnnots is more similar to an object segmentation/annotation task than an attention task, corresponding to a higher rate of roughly 10 cents per annotation per image.

Note that the costs per image for each interface still vary because of different amounts of data collected during a single trial, leading to different numbers of participants required. A summary can be found in Fig. 10.

There is a difference between saliency and importance

Saliency measures which part of an image people look at. Importance measures which elements are most important for understanding its content. Fig. 15 places our attention-tracking methods on the scale from measuring saliency (less intentional) to importance (more intentional). We compare data from interfaces on either end of this spectrum, CodeCharts and ImportAnnots, on both natural images and graphic designs.

The two interfaces produce quite different kinds of data. CodeCharts attention maps reflect common viewing patterns

Figure 15. Interfaces vary in the degree to which they measure saliency (which is more natural) and importance (which is more intentional) of image regions. What BubbleView measures varies slightly depending on whether people are given a direction (D) or allowed to explore the image freely (F).

such as center bias and a focus on localized points of interest, whereas ImportAnnots produces large segmented elements. ImportAnnots and CodeCharts are most similar when the image contains a handful of objects that are both salient and segmentable. An example of where the interfaces coincide and diverge on natural images can be found in Fig. 13. Due to these differences, ImportAnnots and CodeChart heatmaps are only weakly correlated with each other (CC of 0.413 for natural images and 0.491 for graphic designs).

We find specific differences in what is salient versus important in graphic designs (Fig. 14). CodeCharts indicates that workers attend mostly to the center of the image. However, ImportAnnots indicates that workers almost always find text to be important, regardless of its position. Thus, CodeCharts and ImportAnnots are best-aligned when the title is in the center of the design. We explore whether CodeCharts attention maps can be used to recreate importance data on graphic designs with the help of ground-truth bounding boxes for segmented design elements. For each design, we create a ranking of the constituent elements by summing over the heatmap density that falls inside each element, using both CodeCharts and ImportAnnots heatmaps. The two rankings have a Spearman rank correlation of 0.509, indicating that differences in heatmap distribution alone cannot explain the difference in the data produced by these interfaces.

CASE STUDIES

Here we present some ways in which the interfaces from this paper can be used to investigate questions related to human perception and cognition of different image types.

What is attention-capturing in a resume?

Recruiters spend as little as 6 seconds on a resume before making their initial decision [27]. As such, a resume's ability to capture a recruiter's attention at a glance is very important and an apt candidate for analysis by our methodologies.

We use two interfaces to examine resumes. We captured CodeCharts gaze points on 116 resumes with presentation times of 0.5, 1, 2, and 6 seconds to capture temporal and ordering data about the viewing process. We then use ZoomMaps, collected on 29 resumes, to examine what details attract reviewers' further attention. 26 resumes have data for both interfaces, and we have a further 30 resumes for which ImportAnnots are available.

Interface	Use Case	Advantages	Drawbacks
ZM	Capture exploration of large images at multiple scales	Works on images with multi-scale content, natural form of interaction	Coarse approximation of attention
CC	Approximating eye-tracking, esp. for precise viewing durations	Doesn't distort stimuli, experimenter controls timing, fun	Little data per participant, images must fit on screen
IA	Comparing importance of graphic design elements	Produces clean segmentations, captures importance	Not ideal for natural images, measures importance over attention
BV	Approximating eye-tracking	Versatile, cheap	Distorts stimuli and viewing experience

Table 1. Use cases and tradeoffs for each of the four interfaces.

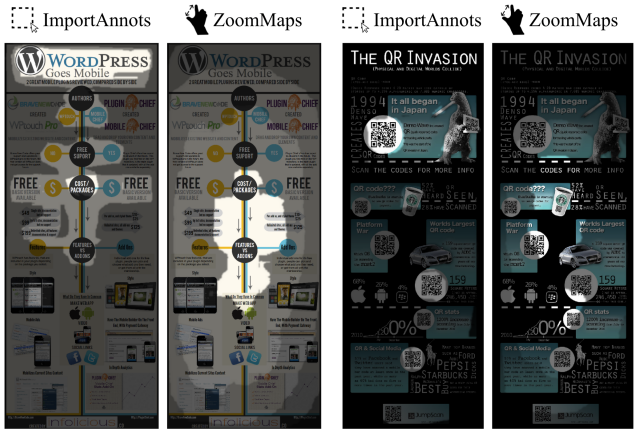


Figure 16. Comparison of ImportAnnots and ZoomMaps on infographics images. Title is always considered as one of the most important elements when using ImportAnnots, but receives little to no zoom-ins on ZoomMaps interface. In addition, QR codes are often labelled as important on ImportAnnots, but are rarely zoomed on probably because they carry no meaning but might be important.

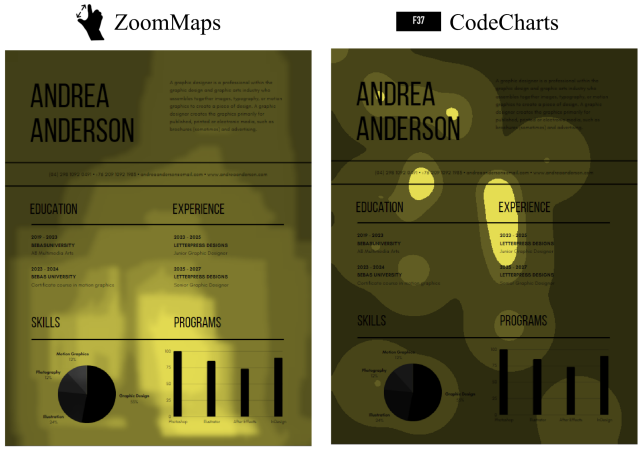


Figure 17. Comparison of ZoomMaps and CodeCharts on resumes. ZoomMaps attention map fails to capture the attention around the title and places a heavier weight around graphical elements.

We find that on 25 out of the 29 resumes analyzed with ZoomMaps, reviewers zoomed in on graphical elements such as bar charts which are often used to represent the skill section. From CodeCharts, we discover that from shorter to longer viewing durations people's gaze moves increasingly towards textual elements, but that gaze moves away from bolded towards non-bolded text (Fig. 18). This suggests that in the first few seconds of viewing reviewers focus on easily-visible elements like titles, section headings, and pictograms. Thus, visualizing resume data may actually be a successful technique for drawing a reviewer's attention to your skills.

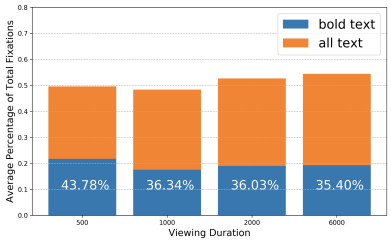


Figure 18. Participants' gaze move towards textual elements at longer viewing duration for 116 CodeCharts images. The white text inside the bar is the % of bold text viewed out of all text. Gaze is more attracted to bold text at earlier viewing duration.

Interestingly, people's gaze continues to be attracted by faces even if those faces are iconographic. Out of 12 resumes that contained a face, all 12 had fixations on the face within two seconds of presentation time. This could indicate that personal portraits on a resume may distract a reviewer from other content.

Finally, we notice once again a difference in what is captured by ZoomMaps, CodeCharts, and ImportAnnots in Fig. 16, Fig. 17 and Fig. 20. Both CodeCharts and ImportAnnots indicate disproportionate attention paid to the title (for CodeCharts, titles attract around 12% of gaze points). However, ZoomMaps almost completely ignores the title as shown in Fig. 17. Out of 368 different zoom windows (excluding ones that contain the entire resume), only 15 focused on the title. None of the average maps produced from ZoomMaps even contain the title. This highlights ZoomMaps' role as providing more fine-grained viewing data on smaller elements.

Figure 19. CodeCharts, ZoomMaps, and ImportAnnots compared to human eye movements on the CAT2000 dataset. CodeCharts best approximates human eye movements. ZoomMaps occasionally focuses on elements in the background instead of picking salient objects out of the foreground, and ImportAnnots specializes in segmenting semantically important elements, often focusing on a single central object. Above each image are the respective NSS scores obtained by comparing attention maps from each of the user interfaces to ground truth eye movements.

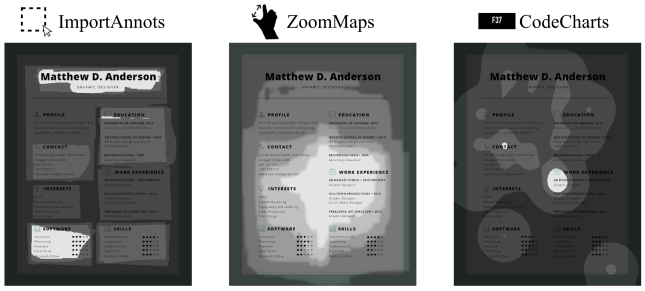


Figure 20. Comparison of 3 interfaces on resumes. ImportAnnots data shows that the title is one of the most important elements. Due to the size of the title, ZoomMaps fail to capture it as an attended-to element, as participants do not need to zoom to view it. Gazes from CodeCharts move around the title in the first 2 seconds and then move away at 6 seconds.

ZoomMaps for academic posters

Additional use cases of ZoomMaps emerge when applied to large visuals such as academic posters. We ran a small in-person study where 8 participants viewed 6 academic posters using the ZoomMaps interface on one day and were asked to answer questions about the poster from memory the next day. Prior to answering the questions for each poster, participants were shown a thumbnail of the poster, roughly 10% of its original size, for 10 seconds. This was too small to be able to read any text but was intended to jog the participants' memory of a particular poster. The thumbnails for 2 of the 6 posters were static and 4 were animated GIFs computed from zoom patterns collected using ZoomMaps.

6 of the 8 participants indicated that they preferred the animated GIFs to the static images and 7 of the 8 participants remembered the posters better when presented with GIFs. This suggests that ZoomMaps data may be useful for generating animated thumbnails to aid in recall.

We also found that, on average, posters were viewed at 4.15 times the resolution at which they originally appear at on the mobile screen. However, some posters had average zoom levels as high as 5.11. Given that all posters were presented at the conference at the same size, the mobile viewing data suggests that the poster with the largest zoom level had a scaling issue: information was presented too small for comfortable viewing. Thus, ZoomMaps can also be viewed as a debugging tool for designs, providing an evaluation of the appropriateness of the physical scale at which information is visually presented.

CONCLUSION

ZoomMaps, CodeCharts, ImportAnnots, and BubbleView provide a variety of options for collecting attention data in a crowdsourced, scalable way. We have demonstrated that they can be used individually to handle particular use cases like collecting attention data on images with multi-scale content that is too big to fit on one screen (ZoomMaps) or gathering saliency data at multiple durations (CodeCharts).

Most importantly, we showed that these interfaces are not simply four redundant ways of collecting the same data. Rather, they fall at different points on the scale of saliency and importance. They can be combined to gain a complex understanding of various aspects of attention like temporal dependencies, variations in scale, and automatic versus considered responses to stimuli. Future work can use these interfaces to study interaction between attention and higher-level cognitive tasks like search, decision-making, and memory.

REFERENCES

1. Roman Bednarik and Markku Tukiainen. 2005. Effects of Display Blurring on the Behavior of Novices and Experts During Program Debugging. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*. ACM, New York, NY, USA, 1204–1207. DOI: http://dx.doi.org/10.1145/1056808.1056877
2. Ali Borji and Laurent Itti. 2015. Cat2000: A large scale fixation dataset for boosting saliency research. *arXiv preprint arXiv:1505.03581* (2015).
3. Michelle A. Borkin, Zoya Bylinskii, Nam Wook Kim, Constance May Bainbridge, Chelsea S. Yeh, Daniel Borkin, Hanspeter Pfister, and Aude Oliva. 2016. Beyond Memorability: Visualization Recognition and Recall. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (Jan 2016), 519–528. DOI: http://dx.doi.org/10.1109/TVCG.2015.2467732
4. Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. 2016. What do different evaluation metrics tell us about saliency models? *CoRR* abs/1604.03605 (2016). http://arxiv.org/abs/1604.03605
5. Zoya Bylinskii, Nam Wook Kim, Peter O'Donovan, Sami Alsheikh, Spandan Madan, Hanspeter Pfister, Fredo Durand, Bryan Russell, and Aaron Hertzmann. 2017. Learning Visual Importance for Graphic Designs and Data Visualizations. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software & Technology (UIST '17)*. ACM. DOI: http://dx.doi.org/10.1145/3126594.3126653
6. Jia Deng, Jonathan Krause, and Li Fei-Fei. 2013. Fine-Grained Crowdsourcing for Fine-Grained Recognition. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*. IEEE Computer Society, Washington, DC, USA, 580–587. DOI: http://dx.doi.org/10.1109/CVPR.2013.81
7. Rachel England. 2018. Twitter uses smart cropping to make image previews more interesting. https//engadget.com/2018/01/25/twitter-uses-smart-cropping-to-make-image-previews-more-interest. (2018).
8. Anonymized for review. 2019. How many glances? Modeling multi-duration saliency. *In submission* (2019).
9. Frédéric Gosselin and Philippe G. Schyns. 2001. Bubbles: a technique to reveal the use of information in recognition tasks. *Vision Research* 41, 17 (2001), 2261 – 2271. DOI: http://dx.doi.org/10.1016/S0042-6989(01)00097-9
10. Qi Guo and Eugene Agichtein. 2010. Towards Predicting Web Searcher Gaze Position from Mouse Movements. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems (CHI EA '10)*. ACM, New York, NY, USA, 3601–3606. DOI: http://dx.doi.org/10.1145/1753846.1754025
11. Jeff Huang, Ryen White, and Georg Buscher. 2012. User See, User Point: Gaze and Cursor Alignment in Web Search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 1341–1350. DOI: http://dx.doi.org/10.1145/2207676.2208591
12. Anthony R. Jansen, Alan F. Blackwell, and Kim Marriott. 2003. A tool for tracking visual attention: The Restricted Focus Viewer. *Behavior Research Methods, Instruments, & Computers* 35, 1 (2003), 57–69. DOI: http://dx.doi.org/10.3758/BF03195497
13. Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. 2015. SALICON: Saliency in Context. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1072–1080. DOI: http://dx.doi.org/10.1109/CVPR.2015.7298710
14. Nam Wook Kim, Zoya Bylinskii, Michelle A Borkin, Krzysztof Z Gajos, Aude Oliva, Fredo Durand, and Hanspeter Pfister. 2017. BubbleView: an interface for crowdsourcing image importance maps and tracking visual attention. *ACM Transactions on Computer-Human Interaction (TOCHI)* 24, 5 (2017), 36.
15. Nam Wook Kim, Zoya Bylinskii, Michelle A. Borkin, Aude Oliva, Krzysztof Z. Gajos, and Hanspeter Pfister. 2015. A Crowdsourced Alternative to Eye-tracking for Visualization Understanding. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 1349–1354. DOI: http://dx.doi.org/10.1145/2702613.2732934
16. Steven Komarov, Katharina Reinecke, and Krzysztof Z Gajos. 2013. Crowdsourcing performance evaluations of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 207–216.
17. Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye Tracking for Everyone. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2176–2184. DOI: http://dx.doi.org/10.1109/CVPR.2016.239
18. George W. McConkie and Keith Rayner. 1975. The span of the effective stimulus during a fixation in reading. *Perception & Psychophysics* 17, 6 (1975), 578–586. DOI: http://dx.doi.org/10.3758/BF03203972
19. Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. 2014. Learning Layouts for Single-Page Graphic Designs. *IEEE Transactions on Visualization and Computer Graphics* 20, 8 (Aug 2014), 1200–1213. DOI:http://dx.doi.org/10.1109/TVCG.2014.48
20. Alexandra Papoutsaki, Patsorn Sangkloy, James Laskey, Nediyana Daskalova, Jeff Huang, and James Hays. 2016. WebGazer: Scalable Webcam Eye Tracking Using User Interactions. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI, 3839–3845.

21. Keith Rayner. 2014. The gaze-contingent moving window in reading: Development and review. *Visual Cognition* 22, 3-4 (2014), 242–258. DOI: http://dx.doi.org/10.1080/13506285.2013.879084

22. Kerry Rodden, Xin Fu, Anne Aula, and Ian Spiro. 2008. Eye-mouse Coordination Patterns on Web Search Results Pages. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems (CHI EA '08)*. ACM, New York, NY, USA, 2997–3002. DOI: http://dx.doi.org/10.1145/1358628.1358797

23. Dmitry Rudoy, Dan B Goldman, Eli Shechtman, and Lihi Zelnik-Manor. 2012. Crowdsourcing gaze data collection. *arXiv preprint arXiv:1204.3367* (2012).

24. Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. 2008. LabelMe: a database and web-based tool for image annotation. *International journal of computer vision* 77, 1-3 (2008), 157–173.

25. Michael Schulte-Mecklenbeck, Ryan O. Murphy, and Florian Hutzler. 2011. Flashlight - Recording Information Acquisition Online. *Comput. Hum. Behav.* 27, 5 (Sept. 2011), 1771–1782. DOI: http://dx.doi.org/10.1016/j.chb.2011.03.004

26. Benjamin W. Tatler, Roland J. Baddeley, and Iain D. Gilchrist. 2005. Visual correlates of fixation selection: effects of scale and time. *Vision Research* 45, 5 (2005), 643 – 659. DOI:http://dx.doi.org/https://doi.org/10.1016/j.visres.2004.09.017

27. TheLadders. 2012. Keeping an eye on recruiter behavior: new study clarifies recruiter decision-making. http://www.bu.edu/com/files/2018/10/TheLadders-EyeTracking-StudyC2.pdf. (2012).

28. Juan Xu, Ming Jiang, Shuo Wang, Mohan S. Kankanhalli, and Qi Zhao. 2014. Predicting human gaze beyond pixels. *Journal of Vision* 14, 1 (2014), 28. DOI: http://dx.doi.org/10.1167/14.1.28