Glance: Rapidly Coding Behavioral Video with the Crowd

Walter S. Lasecki
Computer Science Department
University of Rochester
wlasecki@cs.rochester.edu

Mitchell Gordon
Computer Science Department
University of Rochester
m.gordon@rochester.edu

Danai Koutra
Computer Science Department
Carnegie Mellon University
danai@cs.cmu.edu

Malte F. Jung
Information Science
Cornell University
mfj28@cornell.edu

Steven P. Dow
HCI Institute
Carnegie Mellon University
spdow@cs.cmu.edu

Jeffrey P. Bigham
HCI and LT Institutes
Carnegie Mellon University
jbigham@cs.cmu.edu

ABSTRACT

Behavioral researchers spend considerable amount of time coding video data to systematically extract meaning from subtle human actions and emotions. In this paper, we present Glance, a tool that allows researchers to rapidly query, sample, and analyze large video datasets for behavioral events that are hard to detect automatically. Glance takes advantage of the parallelism available in paid online crowds to interpret natural language queries and then aggregates responses in a summary view of the video data. Glance provides analysts with rapid responses when initially exploring a dataset, and reliable codings when refining an analysis. Our experiments show that Glance can code nearly 50 minutes of video in 5 minutes by recruiting over 60 workers simultaneously, and can get initial feedback to analysts in under 10 seconds for most clips. We present and compare new methods for accurately aggregating the input of multiple workers marking the spans of events in video data, and for measuring the quality of their coding in real-time before a baseline is established by measuring the variance between workers. Glance's rapid responses to natural language queries, feedback regarding question ambiguity and anomalies in the data, and ability to build on prior context in followup queries allow users to have a conversation-like interaction with their data – opening up new possibilities for naturally exploring video data.

Author Keywords
Data analysis; subjective coding; crowdsourcing; video

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation: Misc.

INTRODUCTION

Behavioral video coding provides researchers in the social sciences a lens for studying human interactions [3]. In the domain of HCI, for example, researchers use video analysis to
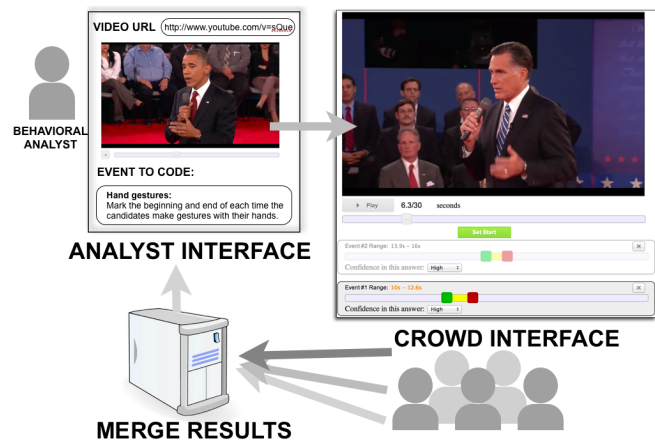
Figure 1. Glance codes behavioral events in video quickly and accurately. When a question is asked, small clips from the video are sent to crowd workers who label events in parallel. The judgments are then quickly merged together and displayed. In this example, we use Glance to count the frequency and duration of eye contact between presidential candidates during a 2012 debate.

study how users interact with computational systems [21, 19] and to develop theories that explain those interactions (e.g., [3, 4]). While video coding affords a systematic measure of behavior, it is notoriously time-consuming. By some accounts, video coding analysis takes 5-10x longer than the play time of video [20]. This does not include the time to develop a reliable coding scheme, to train coders, or to check for inter-rater reliability. Given the high cost and lengthy turnaround required to test a hypothesis, researchers may exhibit sunk-cost reasoning and resist changes to their coding scheme.

Ideally —and maybe eventually —computational approaches will mature to the point of being able to automatically label segments of behavior. However, due to the diverse contexts and subtly interpretive nature of video coding, automated analysis of human behavior currently yields poor results. While commercial video coding tools make this process easier for a research team, they do not eliminate the bottleneck of sequentially viewing all the video [22, 1].

In this paper, we introduce *Glance*, a system that allows researchers to analyze and code events in large video datasets by parallelizing the process across a crowd of online workers (Figure 1). This approach significantly reduces the amount of time required to test a coding hypothesis within video data. Video data can be coded in a fraction of the actual time depending on the size of the available crowd. By distributing the same video segments among multiple workers, Glance can calculate the variance to provide quick feedback result reliability. Glance provides a front-end interface for analysts to enter natural-language queries and to visualize coding results as they arrive.

To evaluate Glance, we coded multiple behavioral events in several different types of video data, including a publicly broadcast political debate and lab-based social science experiment. To investigate the speed of the approach, we coded occurrences of a single type of behavioral event in an hour-long video. Glance produced judgments on a 12 minute-long video in 2 minutes ($6\times$ speedup), and coded 48 minutes of video in just 5 minutes (a nearly $10\times$ speedup).

To evaluate accuracy, we coded for the occurrence and time window of variable-length behavioral events and compared them to a gold standard. Using 10 simultaneous workers for each video segment, Glance was able to accurately identify on average over 99% of the event occurrences and estimated the time window of these events within a 1-second margin of error, in terms of both when they occur and for how long.

By dramatically reducing the speed and increasing the reliability of video coding, researchers have the opportunity to interact more "opportunistically" with their data. Glance's interface visualizes coding results in real-time, allowing an analyst to issue new queries, and cancel or re-frame their original one after seeing initial results. The resulting conversational style of interaction allows video analysts to more quickly explore, develop and refine research hypotheses in ways that are not feasible today.

Our contributions can be summarized as follows:

- We propose a conversational interaction paradigm for video coding that leverages crowd workers with minimal expertise to provide rapid judgments on natural language queries about behavioral events in video data.

- We introduce Glance, a video coding tool for researchers that *(i)* recruits and coordinates an online crowd for behavioral coding, *(ii)* segments video data and distributes video coding tasks to individual crowd workers, and *(iii)* collects and visualizes individual judgments and aggregate statistics in real-time.

- We provide experimental evidence via a large study with Amazon Mechanical Turk workers that the crowd can quickly and accurately identify events in video on-demand.

- We demonstrate an end-to-end interaction using Glance to explore real-world data, and show that the results are quick, accurate, and allow analysts to make informed decisions about subsequent steps in their data exploration process.

ENVISIONED INTERACTION

In this section, we overview an envisioned interaction with a video coding system, which inspired Glance, the system that we introduce in this paper.

Alice is a human-computer interaction researcher who wants to analyze 18 hours of video of computer game play sessions to determine if the system she has built improves players' engagement with one another. She begins by opening her behavioral coding software package and loading the video she recorded during her experiments. Once loaded, she asks the system to find all of the times when at least one person was talking in the video. Instantly, indicators appear below the play bar that mark the 75% of her videos that contain someone talking. She then asks the system to find "when two players are engaged with one another". The system responds with several instances as well as a message indicating that the question is too vague and there is low confidence in the reported results. Realizing that she has underspecified the problem, Alice researches the literature for behavioral cues to look for when measuring engagement, and asks the system to find instances of eight such cues. The system returns separate event logs for all eight behaviors, and from this visualization, Alice is able to see that only five of the behaviors occur more than once in her data.

Given the few returned instances, Alice suspects that those might be only a small part of the two-person interactions in the video. For this reason, she asks the system to find all times where two people talk directly to one another, as a subset of the times when people are talking. The system realizes that this is a follow-up query and only checks sections of the video that it knows has someone talking in them. After seeing the results, Alice realizes her suspicions were correct – only about 60% of conversations were coded for the target engagement metrics. Looking at her initial coding scheme, she realizes that she was only coding for positive engagement cues. After doing another search, she tries asking the system to mark a few negative engagement cues during periods where other cues were not found, but that had two players talking to one another. The system returns a number of examples, and Alice is able to see that, at some points, players are fighting with one another about in-game decisions. In further interactions with the system, she is able to make a number of other corrections to her initial assumptions and develop a well-informed coding scheme that other analysts can be trained on to get reportable data for her research.

Alice's interaction with the behavioral coding tool demonstrates a best-case scenario: the system understands natural language queries, replies instantly, gives feedback on the clarity and expected accuracy of queries even when no ground truth is available, and allows follow-up queries as subsets of previous queries – similar to a Q&A conversation.

In this paper, we present a system, Glance, that attempts to bring analysts closer to this type of ideal interaction by using the crowd to provide highly parallelizable on-demand human intelligence when answering queries. But, before we introduce the tool, we review related work and empirically eval-

uate to what extent the speed and accuracy of crowd-based video coding could support the interaction we envision.

RELATED WORK

Glance is a crowd-powered system that allows users to rapidly iterate on a video coding task. It draws from prior work in both behavioral coding and crowd-powered systems. Prior research in behavioral coding illustrates what would need to be done to approach the envisioned interaction presented in the last section, and work on crowd-powered systems suggests a path for achieving it.

Behavioral Observation and Coding

Observing and coding behavioral events in video is a common research practice in the social sciences, used in fields as diverse as human-computer interaction, linguistics, anthropology, psychotherapy, and child psychology [11, 20]. Systematic observation through video focuses on creating a precise measurement of behavioral streams. To demonstrate rigor, researchers often use "properly trained observers to produce identical protocols, given that they observed the same stream of behavior" [3]. In practice these are often undergraduate students who are given training that may range from a day to multiple weeks for complex events.

Bakeman and Gottman [3] provide a comprehensive introduction into this technique that is largely performed in three steps. First a coding scheme, which defines the set of behavioral categories, has to be developed. This can be done in a bottom-up, top-down approach [34], or, more often, a hybrid approach. In a bottom-up approach the coding scheme is constructed from behavioral features observed during careful viewing and re-viewing of the video, while in a top-down approach it is derived from theory.

The second step is then to train coders. The amount of training required largely depends on the complexity of the coding scheme, and can take up to several weeks or require detailed training curriculum. Training is often seen as a continuous process even after the actual coding has started. For example, Coan and Gottman [11] describe a practice in which coders collectively watch practice videos in order to clarify confusions. We integrate a similar (but shorter) form of training into our tasks to prepare crowd workers.

The final step is then to categorize or "code" the video data according to the coding scheme. At least one independent observer should code some of the videos for a second time to test reliability. Researchers often selectively test inter-rater agreement [12] throughout the process to refine the coding scheme. Practitioners recommend that analysts make several passes through the video data and *code one event type at a time*, in order to avoid missing occurrences [20]. This increases reliability, but also greatly increases the amount of time required to code a video.

Video Coding Tools

Several tools have been developed to support the video coding process, particularly steps two and three described above. For example, ANVIL, Datavyu [1], VACA [9], and VCode [18] all provide an interface for easily annotating audio and video

with event tags, and some allow for testing of inter-rater agreement, *but only* for the *overlap* of observed events. Tools such as GSEQ [4] calculate agreement statistics and provide visual feedback for coders.

Despite this growing availability of video coding tools, there is a lack of support for the highly iterative process of developing, testing, and re-visiting a coding scheme . This process can take several times longer than the length of the video content itself, and as a result, researchers invest considerable time in each iteration of their coding scheme. This makes it challenging for researchers to reanalyze video based on initial findings, to thoroughly explore the different types of events, or to modify a coding scheme based on preliminary analysis. In this paper we address these shortcomings by providing tool-support for a process of rapidly iterating and testing coding schemes via the inherent parallelism of the crowd.

Measuring Agreement

Observer agreement describes the degree to which two or more observers agree with each other [3]. This is different from observer reliability that describes the degree to which an observation aligns with an established "truth", such as a baseline observation protocol. Most observational research only assesses agreement, while reliability is assumed given sufficient agreement. Measuring agreement can: (i) indicate the trustworthiness of observations in the data, and (ii) provide feedback to calibrate observers against each other or against baseline observations. If one assumes the baseline to be "true", then observer agreement can be used to assess reliability. A commonplace statistic to assess observer agreement is Cohens Kappa [12] which corrects for chance agreement between two workers.

Crowdsourcing Video Annotations

Crowdsourcing leverages human computation [32] in the form of open calls to paid online workers from platforms such as Amazon's Mechanical Turk. Crowdsourcing has been used on tasks that rely on human judgment and that are difficult for automated systems. For example, Soylent [7] uses the crowd to edit or shorten writing, VizWiz [8] answers questions about photographs quickly, and Legion [24] follows natural language commands to intelligently control a GUI.

The crowd has also been leveraged in the context of activity recognition systems. For instance, VATIC [33] allows crowd workers to tag where objects appear in a scene. While the crowd provides these annotations, it is not designed to respond quickly to the end-user. Similarly, Legion:AR [25, 28] explores crowd labeling of *low-level* actions in video for assistive home monitoring: workers are asked to watch a video stream as it happens live, and then an automated system labels activities soon after they occur. Unlike our approach, Legion:AR does not process video any faster than an individual can, and is designed for use *not* by a human analyst, but by a Hidden Markov Model-based system. Di Salvo et al. [14] added game elements to an annotation task to get the crowd to mark where an object appears. With Glance, workers identify complex or subtle events, and to accurately identify the range over which they occurred.
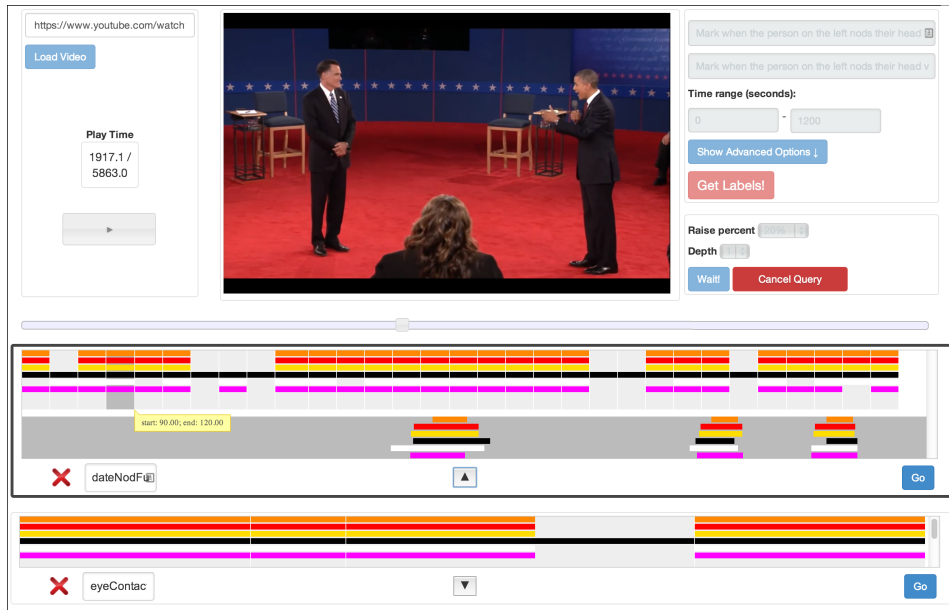
Figure 2. The Glance analyst user interface (AUI). Analysts can load a video from YouTube, ask if or when an event occurs in the video in natural language, and set parameters to control cost and speed. Crowd workers then process this query and return results in a fraction of the playtime of the video. These results are aggregated to simplify the answer for the analyst, and make it more reliable than a single worker's answer.

GLANCE

Glance is a video coding tool that leverages online crowds to produce rapid analyses of video data (Figure 1). To reach envisioned conversation-like interaction with data, Glance allows analysts to ask questions about events in natural language, provides nearly-immediate responses, and supports query refinement by building on known context. We describe the user experience here, then detail the novel technical components in the next sections.

Recruiting a Crowd

When an analyst arrives to Glance's analyst user interface (AUI), shown in Figure 2, the system begins recruiting workers into a retainer so that they can be available within a matter of seconds of posing a query [8, 6]. To load a video, the analyst provides the system a YouTube link, a link to a video on their own server, or directly uploads a file. The video is immediately loaded into the embedded player, and the query creation tool becomes active.

Posing Queries

When the analyst wants to ask Glance a question about the video, they enter a name for the query and a natural language description of the hypothesized behavioral event. To more clearly specify the sought-after event, the analyst has the option to select one or more parts of the video to serve as examples. These example video segments will then be shown to workers during a short training session.

When a query is posted, Glance sends tasks to multiple crowd workers in parallel and retrieves answers as soon as possible. To reduce the cost of these queries, especially during early exploration, where complete coverage of the video might not be necessary, Glance lets analysts select a portion of video

to focus on. The system also allows analysts to adjust the following parameters:

- **Redundancy.** Glance lets the user control the size of the crowd being used (and thus the cost of the query) in part by selecting a level of redundancy for each query. This defines the number of workers who will label each video clip. If the analyst wants to increase the reliability of the response later in the process, Glance lets them increase the level of redundancy – building upon already completed work.
- **Playback Rate.** For some queries, costs can be reduced by increasing the playback rate of the video. While this is also common practice for current video coding systems, combining it with the parallelism of the crowd can lead to even quicker responses. As increasing playback speed is not practical with all types of data and events, we leave it to analysts to know when it is appropriate to use.
- **Sampling Rate.** To get an initial idea of what types of events occur in a video, Glance lets analysts select a video clip sampling rate. This allows analysts to see how frequently their events arise, without needing to hire workers for a complete run. Glance selects this percentage of clips uniformly across the analyst-specified range. If analysts want to take a closer look at their data later, Glance lets them increase the sampling rate without repeating work that has already been done.
- **"Gist" Mode.** In "gist" mode, the system asks workers to simply mark if any instance of the event occurs within a the clip, rather than asking them to mark the exact range in which it occurs. This option provides a means for quickly illustrating the high-level frequency of pertinent events.

Each of these parameters lets the analyst fully specify the limitations and behavior of the system for a specific query. Note that this additional complexity is entirely voluntary – analysts
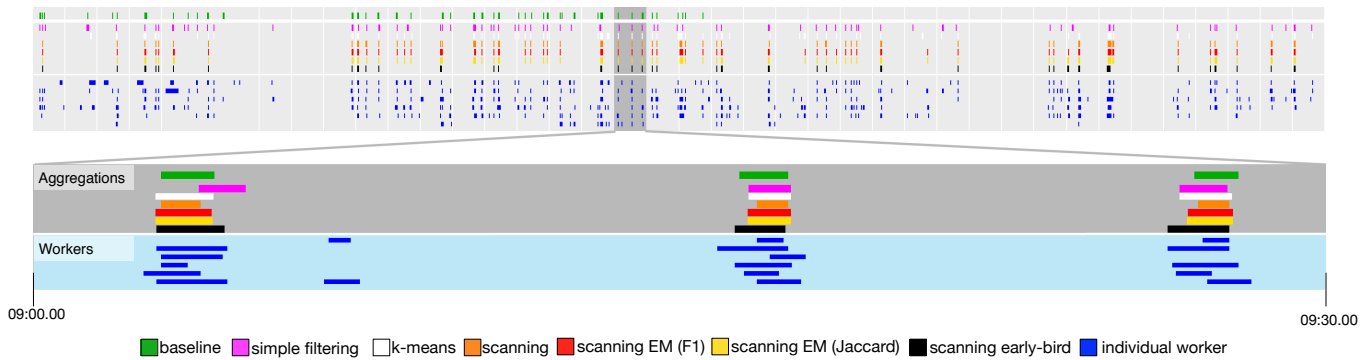
Figure 3. A visualization of crowd worker responses when coding "head nodding" events in a 20-minute video. Colored lines denote the marked span of an instance of the event. The top half of the figure shows the results for all 20 minutes, and the bottom half shows a zoomed-in version of one 30-second clip. Worker inputs are shown at the bottom of the zoomed clip, while the aggregated spans produced by our 6 schemes are shown above.

may use the built-in defaults for any or all of these settings if they so choose.

Visualizing Results

Once a worker codes a video clip, the judgement gets displayed in the video coding UI. Glance visualizes partial results to show analysts the work-in-progress results. In order to reduce the complexity inherent in multiple worker judgements, Glance aggregates the answers. Analysts may choose to use one or more aggregation algorithms, and may choose to see raw worker input as well. Figure 3 shows an example visualization and the next section provides details on the aggregation algorithms.

The problem of visualizing results that come in over time is one that has been studied in data analytics as well (e.g., [5]). There, the bound is the speed of machine computation over large datasets, which parallels the human computation delay seen in Glance, or any crowd-powered system.

Receiving Feedback on the Query

The reliability of the resulting worker judgements is understandably tied to the description given by the analyst. To help guide analysts, Glance provides feedback about a query's clarity and likelihood of convergence. The system analyzes variance between workers and predicts the quality of results, even before a query is finished, and even when there is no baseline data for comparison.

Glance can also highlight when individual clips may contain an exception to an otherwise stable rule (e.g., highlighting when one worker may have either made an error or noticed something the rest of the crowd did not). The next section details how Glance facilitates this feedback by observing agreement levels between workers.

Refining a Query with Followup Questions

To further support conversation-like interaction with video data, Glance supports followup queries that use prior responses as filters to reduce the number of video clips under consideration. For example, an analyst can search for the existence of a broad category of behavior, and then — only within clips that have this behavior — ask for more specific variations of that behavior. To select a filter query, the analyst

can select one or more previously completed queries, and the new query will only be run over positive instances of the returned clips. Conversely, the 'inverse' selection tells the system to only run a query in selected clips where a previously coded event was *not* found.

In sum, Glance supports conversation-like interaction with video data by accepting natural language queries, responding quickly, providing feedback when queries are too vague or when anomalies occur, and supporting a "layering" approach to queries that build on prior coding results. The next section describes three empirical studies that demonstrate the technical innovations in Glance.

FEASIBILITY EVALUATIONS

To further describe and validate the core features of Glance, we ran a series of feasibility experiments. The following three sections provide empirical studies for three core aspects of the Glance system: *(i)* the time required to respond to natural language queries, *(ii)* the accuracy of multiple workers in aggregate, and *(iii)* the feedback provided to analysts to improve their natural language queries based on incoming coder agreement. First, we describe the sample data, participants, and procedures used for these experiments.

Method

Sample Data
We evaluated the core components of Glance using video clips from two different data sets:

- **2012 presidential debates.** We leverage the first hour of video from the presidential town hall debate between Barack Obama and Mitt Romney. Presidential debates have long been subject to scientific analysis, examining aspects such as visual camera treatment [31], gesture analysis [10], and social media reactions to debate dynamics [30, 15]. The video shows the two candidates interacting with each other, audience members, and the moderator.
  Across our studies, Glance coded for five different types of behavioral events. Workers were instructed to mark the spans of time when the candidates: 1) made eye contact with each other, 2) argued with each other, 3) transitioned from sitting to standing or vice versa, 4) exhibited rapid

changes in mood, and 5) used hand gestures. To establish a baseline for the purposes of quality comparisons, 2 researchers from our team independently hand-coded a portion of the overall video (5-minute clips for the first 4 events, and a 10-minute clip for the 5th event), then came to a consensus on each event.

- **Design team interactions.** We selected a second data set from a human-subjects experiment where two people worked together on a design task and provide feedback to each other on their designs [16]. The data comprise 30 seconds of interaction with 10 different pairs of people; all participants provided written permission to be included in this study. We kept these clips short, similar to Gottman's 'thin-slice' studies of married couples [2]. Across our studies, Glance coded for four different behavior events: Workers marked segments of video when the designers 1) made eye contact, 2) shifted their focus between different designs, 3) provided positive feedback, and 4) displayed a burst of excitement. To establish a baseline for quality comparisons, our research team independently coded these same four events in all ten clips.

These two data sets are representative of data typically analyzed using video coding. They are sufficiently large and complex to allow us to confidently evaluate our approach.

Setup and procedures
Participants: Across the various studies, we recruited a total of 493 unique workers from Mechanical Turk. In total, these workers marked 2593 clips.

Training: When a worker arrives to the task, she is first shown a brief tutorial on the general coding task she will be asked to perform, and then asked to complete an interactive tutorial that confirms she understands what to do, or provides feedback until she completes the sample task correctly. At this point, the worker remains in the retainer model until a query gets posted.

Task: When an analyst posts a query, workers in the retainer pool first see a brief description of the event to be coded, and an example, if the analyst specified one. Then they proceed to Glance's worker UI (Figure 4) where they see a video clip and a button that lets them mark the beginning and end of behavioral events. As they watch the video, they can press this button and then a new slider will appear below the video that marks the beginning of the event and follows the current position in the video until they click the button again to mark the end of a behavioral event. Workers can then go back and adjust times if corrections are needed. Workers can also mark as many or as few event spans as they see fit.

In 'gist' mode, the worker UI is simplified to elicit only whether the event has occurred or not. As soon as a worker marks the presence of an event and hits submit, the worker gets a new clip and the data gets recorded and displayed in the analyst's UI.

Study 1: Eliciting Rapid Results
The first of three feasibility studies focuses on how rapidly Glance can respond to natural language queries. For this evaluation, we ran several preliminary time trials, as well as, a live real-time experiment at scale.
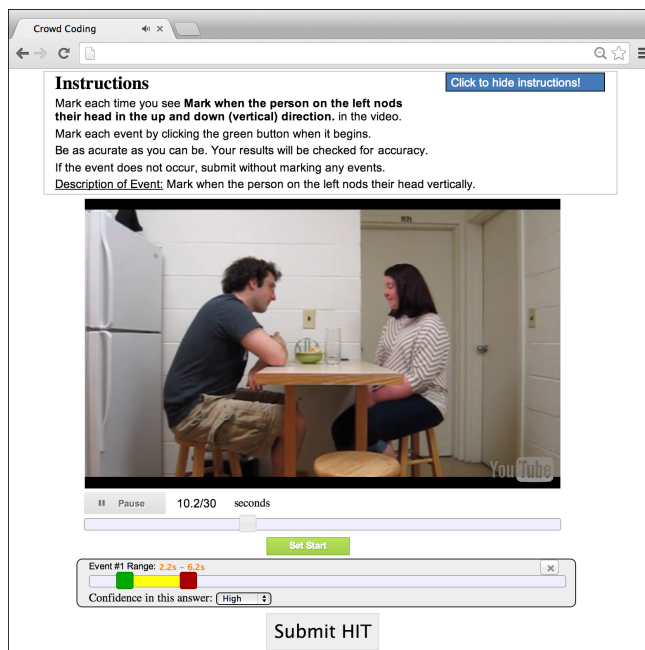

Figure 4. Glance's worker interface. Workers are asked to press a button when they see an event start or end to get an initial estimate of the time range of the event. Workers can then correct the marked range if needed by sliding the start or end marker. In "gist" mode, workers are only asked to mark whether or not they saw the event occur at all.

Time trials
We measured the average time that it took workers to view and mark events in 30-second clips from the debate video. In this trial, we had ten workers per clip code one of four different behavioral events (eye contact, arguments, sit/stand transitions, and mode shifts. It took an average of 60.05 seconds ($\sigma = 12.29$) per clip to mark the start and end times of all events. On average, there were 0.93 events per clip.

We also tested the "gist" mode where workers provide a binary answer — instead of a time span – as soon as they perceive a behavioral event. This *drastically* reduced the coding time — by nearly $10\times$ — by removing the need to watch the entire clip and to fine-tune the event markers. The same 30-second clips of the debate videos took ten workers an average of 8.7 seconds ($p < .01$).

To get even faster results, we can increase the video playback rate. When we increase the playback rate ten-fold, we get a significant improvement from 8.7 to 4.4 seconds ($p < .05$). We further explored how playback speed and worker redundancy affected the time to detect an event (Figure 5). In general, increasing the playback speed reduces the time to return coded events, but sees diminishing value above 2x speed. Similarly, if an analyst wants to increase confidence by hiring multiple workers per clip, this increases the coding time.

Increasing playback speed is also a common technique in traditional video coding. Other traditional approaches for reducing latency can also apply to crowd video coding, although some may require more training.
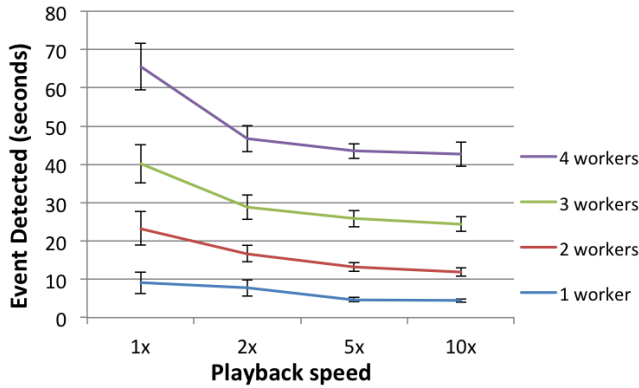
Figure 5. As playback speed is increased, there is a non-linear decrease in the response rate of the first N workers (gist mode).
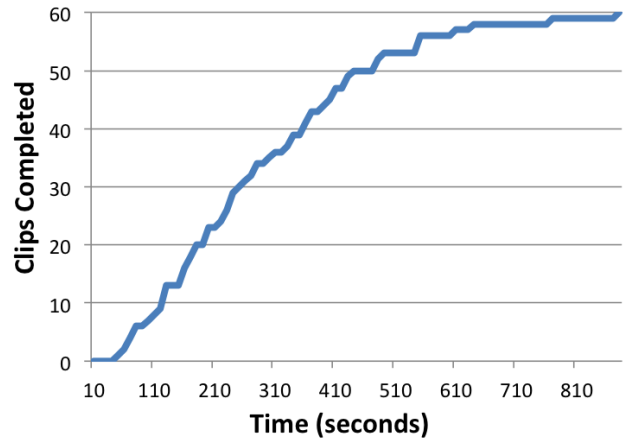

Figure 6. A plot of the number of 1-minute clips from an hour-long video being completed by crowd workers in real-time as part of our live trial. In two minutes, 20% of the content was labeled. In five minutes, 80% of the 60 clips were labeled. This demonstrates that large groups of workers (> 50 people in all in this case) can be recruited simultaneously to complete our coding task quickly.

Coding an hour of video in real-time

To understand how Glance performs on a real video coding task, we coded an entire hour of video from the presidential debate as sixty one-minute clips. We set the playback rate to $2\times$ speed and performed a full coding of the start and end times for all occurrences of eye contact between the two candidates. To optimize for speed, we recruited roughly 50 workers into a retainer. Recruiting workers using a retainer model [8, 6] allows Glance to get workers started on a task in as little as a second.

Figure 6 shows the amount of time required to complete all 60 clips. After about 60 seconds, while the first workers view the video and mark events, answers begin to arrive. In the first 5 minutes after submitting the query, 48 minutes of the video (80%) had been fully coded. As the task nears completion, we see a decrease in the completion rate, in part due to having fewer workers in the retainer at that point. These results confirm that it is possible to code large amounts of video very quickly, especially if enough workers are recruited (in this case, we had just under 50 workers for the first 48 clips).

Study 2: Aggregating Results

The second feasibility study focuses on the accuracy of multiple workers when aggregated. Through crowdsourcing and massive parallelism, Glance can quickly produce answers to queries. However, with multiple workers coding the single clip, we observe different levels of agreement, due to the specificity/ambiguity of queries and worker differences (Figure 9). Individual crowd workers could miss events or misunderstand the query. Some workers put no effort into tasks. To conceal these details and give analysts a clear indication of the occurrences of behavioral events, Glance aggregates worker responses using an *input mediator*.

Glance uses a variety of input mediators to fit the needs of analysts (e.g., favoring recall over precision, or visa versa). To explore the space of mediation strategies, we implemented six input mediators for illustrating the consensus opinion on each clip:

1. **Simple Filtering.** Our first approach uses a set of heuristic filters to disambiguate worker input by removing outliers (mid-points or spans more than 2 standard deviations from the mean), then determines the mode of the numbers of segments marked in a clip and averages the start and end points of responses from workers that match the correct number of segments marked. To avoid discarding input from workers who saw the same basic events, but marked them slightly differently (e.g., as two sequential events instead of a single continuous one) we also look at input where segments are subsumed by the majority of other workers with the correct number of inputs.

2. k-**means Clustering.** To move beyond basic overlap detection, we implemented a k-means clustering algorithm that groups the start and end times of k clusters. To select k, we use the mode number of events marked by workers, after combining all majority-subsumed events into single spans (as described for filtering). Once worker segments are clustered, the final answer is generated by averaging all of the spans in each cluster and defining an equivalent range around the cluster's centroid.

3. **Scanning.** As opposed to trying to aggregate their marked spans directly, this approach uses a "scanning" pass to determine if the majority of workers agreed that an event occurred at that point in time. To calculate this, we discretize the clip into 10-millisecond bins and check for agreement within each bin. To prevent small coincidental gaps in majority agreement from fracturing events into multiple instances, we set a 0.5 second minimum bound for how small the gap can be between two events for them to be considered distinct.

4. **Scanning with E-M on the F1 Score.** The Scanning approach tended to produce a bias for shorter spans (the core parts of a behavioral event where most workers agree), and often single events were split into multiple because of coincidental gaps in coverage that caused the agreement to fall below majority during a small span of time. To counteract this, we implemented a new mediator that uses the initial scanning pass as a starting point, and then runs a version of Expectation-Maximization proposed by Dawid and Skene [13] to improve the initial guess for the seg-
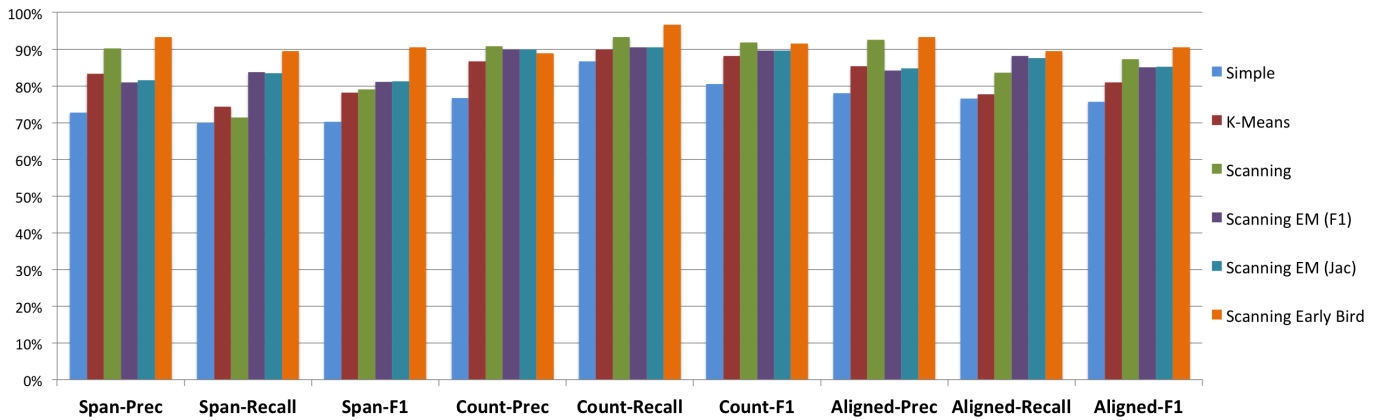
Figure 7. Precision, recall, and F1 score results for all 6 of the worker aggregation schemes we explore. Using our "scanning" approach with additional filtering and adjustments proves the most effective overall, and is significantly better than simple filtering or k-means clustering ($p < .05$).

ment with multiple observers. Specifically, the following two steps occur iteratively: (a) the workers are assigned weights that correspond to our *confidence* in their answers and (b) the start and end point of the best-guess segment are updated by computing the weighted average of the workers' start and end points, respectively. Our confidence in a worker corresponds to the agreement between her individual answer and the current best guess for the segment. In this approach, we use the F1 score[1], where the answer of a worker is tested against the current best guess for the segment. We repeat these two steps until the method converges —when the weights (confidence) of the workers do not change, or equivalently, the best guess for the segment remains the same.

5. **Scanning with E-M on the Jaccard Index.** This approach is similar to the one above, but runs Expectation-Maximization using the Jaccard index between the worker's answer and the current best guess as the confidence level in a worker.

6. **Scanning with Filtering and Early-Bird Bias.** We found that both of the EM modifications to scanning typically resulted in over-sized spans for events. The recall improved, but at the cost of precision. As a result, we looked at other potential improvements to the basic scanning method. Due to a natural latency in human response, the scanning methods tend to mark the onset of behavioral events a bit late (the observed start time was often behind the true time). To counteract this, we created a mediator that uses scanning to find the bins where a majority agreed, but then it shifts the start time earlier. We calculate a time shift using the average of the earliest start time reported by a worker and the aggregate start time across all workers. We use the midpoint because the earliest worker often observed subtle cues that let them anticipate the event early. To prevent this use of a single response from adding potentially far-fetched answers to the final response, we remove all starting times that are outliers from this selection process.

To compare our input mediators, we calculated precision, recall, and F1 score using three different strategies: 1) agree-

ment of the occurrence time and *span*, 2) agreement in the number of behavioral occurrences *count*, and 3) agreement of the span after running multiple sequence alignment to correct for any slight misalignment that may have otherwise produced low precision and recall (*aligned*). As Figure 7 shows, our modified scanning approach was the most effective overall. It was significantly better than the simple filtering and k-means approaches ($p < .01$), and borderline significantly better than the basic scanning approach ($p = .05$). Basic scanning did not perform significantly better than the scanning with EM approaches.

Study 3: Giving Feedback by Measuring Agreement

The third feasibility study focuses on how Glance provides feedback to help analysts improve their natural language queries based on emerging coder agreement and potential edge cases in the data. This feature stems from the realization that not all queries are created equal. Since analysts specify their queries to Glance in natural language, their descriptions could be vague or underspecified. Queries are "ambiguous" when their lack of specificity leaves them open to multiple interpretations by workers. In contrast, we refer to well-defined queries as "concrete". Of course, every query falls on a continuum between these two classifications, and generally speaking, as analysts develop a coding scheme, their queries will become less ambiguous.

In this section, we explore how we can give analysts feedback on the clarity of their query, detect edge cases in descriptions, and even predict the quality of crowd responses without requiring baseline values.

Detecting Agreement

To give analysts feedback on how well workers understand their query, and how good the resulting responses are likely to be, we measure the level of agreement between workers. As a preliminary trial, we compared the total count occurrence between workers when coding two different descriptions within the same clip, concrete ("the person is leaning forward") versus ambiguous ("the person is engaged"). From this trial, we find that the ambiguous description led to significantly more disagreement among workers in the total count of behavioral events ($p < .01$). Further, we see an obvious difference in
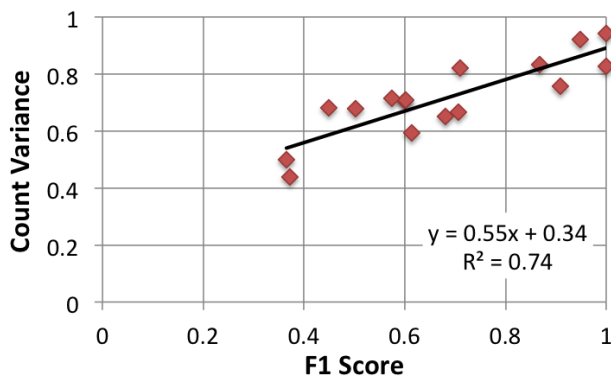
---

[1]We also tried using the precision and recall individually to define the confidence in each worker, but the F1 score —which appropriately combines the two metrics —works better.

Figure 8. **Correlation between our agreement score (variance in the number of segments marked by workers in a single clip) and the aligned F1 score ($R^2 = .74$). This suggests that the level of agreement between workers can be used as a predictor of the overall answer quality even when no baseline is available to compare to.**
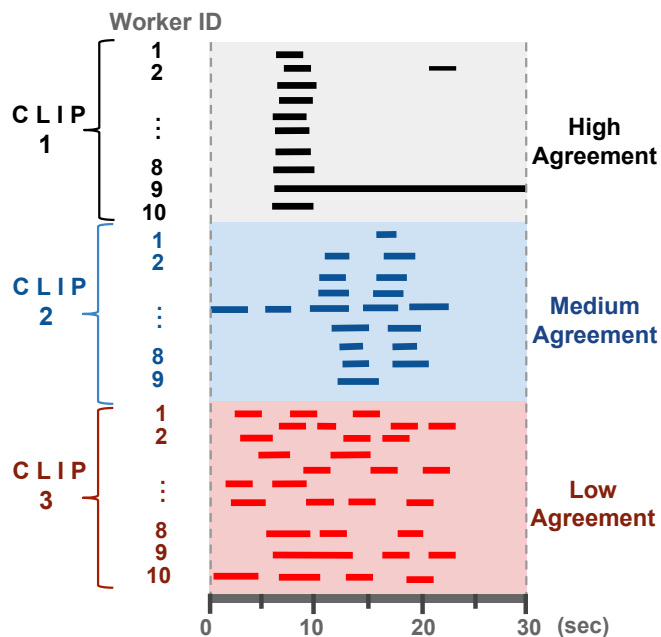

Figure 9. **Visualization of the agreement levels between workers for three 30-second clips (in black, blue, red, respectively). The Y-axis is the worker ID and the X-axis is time (position). Each bar (segment) represents an event marked in the video clip.**

the variance of concrete events ($\sigma^2 = 0.014$) versus more ambiguous events ($\sigma^2 = 0.884$) (Figure 9). Intuitively, this makes sense – when subjectively coding an ambiguous event, different workers may have different interpretations. This can be avoided if analysts provide clear descriptions of events, and one (or more) examples to workers.

To turn this insight into a means of providing feedback to analysts, we created two different agreement metrics. We look at how well each agreement measure predicts the F1 score of the aligned metric described in the previous section. The aligned metric provides a good measure of overall fitness due to the fact that it tolerates small time shifts in the marked segments.

- **Fleiss' Kappa:** A common way to measure agreement level is to use a kappa score to measure inter-rater reliability [17]. When more than two coders are present, Fleiss' kappa can be used. However, Fleiss' kappa assumes that fixed, discrete information is being coded (such as whether a paragraph of text is written in a hostile tone or not). To adapt this measure to our continuous domain, we again bin time into $10ms$ units, then determine if each worker is implicitly contributing a 'yes' or 'no' vote for the event occurring in that bin (i.e., 'yes' if they marked that bin in one of their event spans, and 'no' otherwise). Despite its roots in existing inter-rater reliability settings, it is not a strong predictor of the final accuracy of a given set of inputs. Comparing to the aligned F1 score[2], we find $R^2 = .24$.
- **Count Variance:** Another important way to measure agreement is the overall variance of the different worker inputs. We explored using the variance in the number of events workers marked in a segment (count), total duration of the segments they marked (span), and area of each segment marked (area), as well as combinations of the three. Most combinations scored between $R^2 = .3$ and $R^2 = .5$. In the end, measuring the count variance was the most predictive of the F1 score of the whole clip, with a Pearson correlation of 0.86, $R^2 = .74$ (Figure 8).

---

[2]We use the aligned F1 score because it is most indicative of the overall quality of the final answer.

While different in their predictive power, the two measures did support one another's findings. The count variance tightly correlates with the Fleiss' kappa, with a Pearson correlation of 0.87, and $R^2 = 0.76$ (Figure 10).

Threshold Pruning
Our ultimate goal in measuring agreement is to use it as a proxy for performance that can be calculated immediately, without the need for a baseline to be calculated (which would defeat the point of having a rapid-response tool!) The trend observed when plotting F1 score against agreement score (Figure 8) indicates that the count variance is a good predictor. But does it really work?

The results shown in Figure 7 have been filtered by removing low-agreement clips. To compare the clips in terms of workers' agreement level (response), we performed ANOVA analysis with the clip ID as a single factor with 10 levels. We found that at least one clip had a significantly different mean agreement level from the other sample means ($F_{9,55} = 4.4555, p < .001$). To understand how the mean agreement levels differ, we performed multiple pairwise comparisons using Tukey-Kramer HSD, to correct for Type I errors. At the 95% confidence level, we found that the highest mean agreement value was statistically significantly different from the mean agreement values of four other sessions. We used this as a threshold value (an agreement level of 0.72) and filtered out the three significantly lower trials. This resulted in a 7.8% to 11.1% average increase in the performance of each aggregation method.

In practice, we will not have such a wide range of results to draw from when selecting a cutoff. Thus, we won't be able to predict an exact threshold a priori. However, our goal
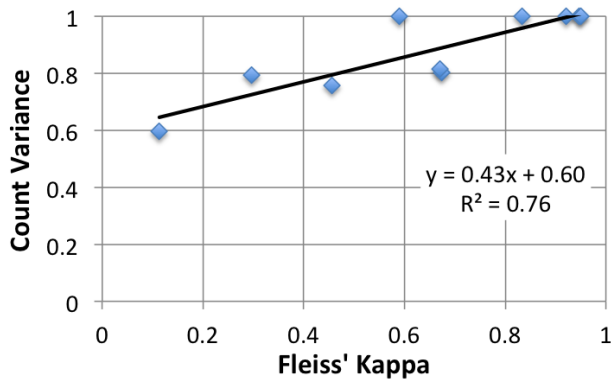
Figure 10. While the count variance is much more predictive of the final F1 score of a response ($R^2 = .74$ versus $R^2 = .24$ for Fleiss' kappa), the count variance still tightly correlates with Fleiss' kappa, suggesting they are measuring similar aspects of agreement.

when providing agreement as feedback to analysts is to provide some signal that they can use to determine how confident they can expect to be in Glance's results. In high-agreement cases, Glance's precision and recall exceeded 99%.

Giving Feedback
The correlation and corresponding effectiveness of filtering based on differences in agreement levels denote it is an effective indicator of the final quality of the answers to a query. This in turn suggests that we can provide this measure to analysts to allow them to judge how consistently workers are agreeing with one another, and thus how accurate they can expect their results to be. For particularly low values of responses for a query (we use below 0.50 in practice), Glance can even alert users to the potential issues with their query. Most commonly, low values will occur when very subjective descriptions are provided because each worker will interpret the event they are looking for slightly differently. As results arrive, this measure can instantly be applied, potentially allowing analysts to retract their query before it is completed in the event that it is unclear to workers – something not possible using current approaches where results are typically not seen until after they are completed.

Even when the overall agreement score is acceptably high, Glance can detect when a subsection of the video (even a single clip) falls below the expected norm. In this case, it is likely true that something in that video is causing confusion for workers (e.g., an edge case that the analyst did not think or know to clarify for in the initial description). In these cases, the clip can be highlighted and shown to analysts who can choose to skip the clip, redefine the query, or continue.

EXAMPLE SCENARIO AND EVALUATION
To demonstrate Glance and the AUI, we present an example interaction between an analyst and Glance. Throughout this example, we report *real* data coded by the crowd. This data came from a third video dataset showing a couple on a date, which is 20 minutes long and includes baseline data for 4 events for the entire video, resulting in 80 minutes of video content. This video is separate from the one used in the evaluations of the individual approaches above.

Our imaginary analyst is a relationship researcher interested in quickly coding a video of a couple dating. Since our analyst initially does not even know what she is most interested in or if the data is valid, she decides to run a rapid-response task with no redundancy but 100% coverage to determine when the two people are talking to one another. This query takes an average of 9.6 seconds to return results, and correctly identifies 100% of the 35 segments in the video that have conversation. The same query with a redundancy level of three would have still only taken 20.6 seconds on average.

Once our analyst sees that there are enough instances of conversation between the two partners to make the video potentially interesting, she decides to ask a follow-up question: "when are the people in the video engaged?" This question is asked about only the 35 clips that contain interaction between the two people in the video since it is a follow-up. After just 5 clips are completed, the system alerts our analyst that worker agreement on the answer is potentially very low. She sees that the agreement score is just 0.57 (recall that our threshold value was 0.72 before), so she decides to stop the query early and rephrase her question to be more clear.

In an attempt to make her query less ambiguous, she asks workers to code specific actions instead of broad interaction properties. For her first query, she asks "when is the person on the right laughing?", of which the crowd finds 71.0% of the 53 instances with 99.9% precision. For her second query, she asks "when is the person on the left nodding?", of which the crowd finds 64.6% of the 87 instances with 99.2% precision.

To check for negative signs during the interaction, she then asks the crowd to find any instances where someone is checking their phone during the date. Because she is mainly interested to find out if this happens a notable number of times, she turns on the gist mode for a quick response. She soon finds that Glance correctly identified 100% of the 13 instances of someone checking their phone. Based on the high amount of direct interaction, our researcher begins to create a final coding scheme that best fits her data.

DISCUSSION
Glance's quick, conversation-style interaction goes beyond the paradigm adhered to by prior work, which assumed that making coding faster, even for data exploration, was something that must be done while focusing on a single human in the loop. Glance uses the parallelism available via the crowd to not only greatly improve the speed at which video data can be explored by analysts while effectively compensating for the added complexity of getting multiple results, but actually uses the presence of multiple overlapping contributors to predict the effectiveness of the codes based on their agreement. But there are many challenges involved in robustly supporting analysis of behavioral video.

Query Ambiguity
Our results suggest that the quality of worker responses is highly dependent on the level of ambiguity in analysts' queries. While this is a drawback, it is a bias shared with existing approaches that involve human coders. However, with shorter training times and more exploratory coding schemes

compared to the well-evaluated ones used in traditional approaches, where getting coding data is very time consuming and expensive, Glance may help analysts better understand the ambiguity of their queries and refine their coding scheme and instructions to compensate.

Context

Maintaining context for workers, who may need to know information from prior clips to accurately understand an event's context within a task, is also a key issue when splitting tasks among multiple people. The general case of this problem, where information from any previous point in the video may provide the necessary context for workers, requires novel techniques and is beyond the scope of this paper. However, the current version of Glance does try to ensure that individual events are captured in their entirety in a single video. To do this, we can look at workers who label events that continue to the end of a clip or begin immediately, allowing for some small ϵ of error. In our experiments, less than a second was enough to capture all instances where an event was split, but this type of division happened so little in our experiments that no generalizable rule could be devised. Situations where information from more than one clip away was needed to code and event never arose, largely due to the types of questions we were asking in our examples.

Worker Training

We do not claim that Glance can code every type of event. For coding systems that are highly granular, and that capture very subtle events coders would normally be trained for periods up to weeks to be proficient enough to complete their task reliably. Crowd based coding has the potential to reduce training efforts, particularly for complex coding schemes, as each worker only has to learn a subset of the codes rather than the entire coding system.

We currently use two forms of on-the-spot training to prepare workers for tasks: (i) a general interactive tutorial that gives workers feedback on an example task until they can complete it successfully (given before workers are sent to the retainer pool), and (ii) a dynamically created tutorial for the specific event being coded that consists of a text description and one or more video examples, if provided by the analyst. However, this does not fully leverage prior work that has shown workers remember task details from one session to the next [29], potentially facilitating long-term learning, but instead presents a tool for using crowd workers' common knowledge to code events now, while future work will explore how to train interested crowd workers over time to allow them to handle increasingly complex tasks.

Cost

Workers were paid $0.15 USD per task to code 30 seconds of video, which took them one minute on average. This equates to a $9/hour effective rate, which is comparable to a work study student. In some scenarios, video is coded redundantly. At a depth of 5 workers, which we do not believe to be necessary in practice, the cost per minute could be as high as $2 \times 5 \times \$0.15 = \$1.50$/minute. The potential advantage of Glance comes from the ability to get feedback more quickly, which means that money is less likely to be wasted on coding that is not used or that has to be changed later. Therefore, total coding costs could be less.

FUTURE WORK

Glance is still only the first step towards allowing analysts to truly have a "conversation with their data". Latency can be further improved by integrating automated coding where possible. By using the crowd to make it possible to handle these natural language requests now, realistic data on how users pose and refine questions can be collected and used to train automated systems to both elicit clear queries and recognize events more accurately.

For longer, more in-depth tasks, context can be maintained more robustly through crowd curation of a shared "memory", and a richer interaction with the system is possible by allowing workers to respond to queries (individually or as a collective) when something is unclear [23, 27]. This is similar to other types of message passing that has been observed as an effective way to boost problem-solving abilities in crowdsourcing before [35]. This might also lead to cases of serendipitous discovery of new information, as has been observed with crowd-powered conversational search tasks [23].

We will be releasing this tool for open use to both allow analysts access to the coding tool to advance their own work, as well as to collect data on how our conversational interaction paradigm is used in real settings to code video. Ongoing work will explore how the potential for privacy threats (see [26]) might be mitigated using video filtering techniques that allow for recognition of the queried events, without revealing the identity of people in the video.

CONCLUSION

In this paper, we have presented the motivation, design, and evaluation of Glance, a crowd-powered system for rapidly coding video. We have shown that by incorporating the effort of many human workers, the system is able to analyze video very quickly, making possible a conversation-like interaction with data that was not previously feasible. To further support iterative refinement and validation of hypotheses on video data sets, we have introduced methods that use variance in crowd responses to support iterative improvement of coding schemes, and introduced aggregation strategies that help produce reliable codes over time span inputs. Glance dramatically reduces the time required for the arduous and slow task of video coding, and in the process may change how social scientists do their work.

REFERENCES

1. Datavyu. http://datavyu.org/.

2. Ambady, N., and Rosenthal, R. Thin slices of expressive behavior as predictors of interpersonal consequences: A meta-analysis. *Psychol bull 111*, 2 (1992), 256.

3. Bakeman, R., and Gottman, J. M. *Observing interaction: An introduction to sequential analysis*. Cambridge University Press, 1986.

4. Bakeman, R., and Quera, V. *Sequential Analysis and Observational Methods for the Behavioral Sciences*. Cambridge University Press, 2011.

5. Barnett, M., Chandramouli, B., DeLine, R., Drucker, S., Fisher, D., Goldstein, J., Morrison, P., and Platt, J. Stat!: An interactive analytics environment for big data. In *SIGMOD 2013* (2013), 1013–1016.

6. Bernstein, M. S., Brandt, J. R., Miller, R. C., and Karger, D. R. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *UIST* (2011), 33–42.

7. Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., Crowell, D., and Panovich, K. Soylent: a word processor with a crowd inside. In *UIST* (2010), 313–322.

8. Bigham, J. P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R. C., Miller, R., Tatarowicz, A., White, B., White, S., and Yeh, T. Vizwiz: nearly real-time answers to visual questions. In *UIST* (2010), 333–342.

9. Burr, B. Vaca: A tool for qualitative video analysis. In *SIGCHI EA* (2006), 622–627.

10. Casasanto, D., and Jasmin, K. Good and bad in the hands of politicians: Spontaneous gestures during positive and negative speech. *PLoS One 5*, 7 (2010), e11805.

11. Coan, J. A., and Gottman, J. M. *Handbook of Emotion Elicitation and Assessment*. Series in Affective Science. Oxford University Press, 2007, ch. The Specific Affect Coding System (SPAFF).

12. Cohen, J. A Coefficient of Agreement for Nominal Scales. *Educ Psychol Meas 20*, 1 (1960), 37.

13. Dawid, A. P., and Skene, A. M. Maximum likelihood estimation of observer error-rates using the em algorithm. In *Appl Stat - J Roy St C* (1979), 20–28.

14. Di Salvo, R., Giordano, D., and Kavasidis, I. A crowdsourcing approach to support video annotation. In *VIGTA* (2013), 8:1–8:6.

15. Diakopoulos, N. A., and Shamma, D. A. Characterizing debate performance via aggregated twitter sentiment. In *SIGCHI* (2010), 1195–1198.

16. Dow, S., Fortuna, J., Schwartz, D., Altringer, B., Schwartz, D., and Klemmer, S. Prototyping dynamics: Sharing multiple designs improves exploration, group rapport, and results. In *SIGCHI* (2011), 2807–2816.

17. Fleiss, J. L. Measuring nominal scale agreement among many raters. *Psychol Bull 76*, 5 (1971), 378–382.

18. Hagedorn, J., Hailpern, J., and Karahalios, K. G. Vcode and vdata: Illustrating a new framework for supporting the video annotation workflow. In *AVI* (2008), 317–321.

19. Hailpern, J., Karahalios, K., Halle, J., DeThorne, L., and Coletto, M.-K. A3: A coding guideline for hci+autism research using video annotation. In *ASSETS* (2008), 11–18.

20. Heyman, R. E., Lorber, M. F., Eddy, J. M., West, T., Reis, E. H. T., and Judd, C. M. *Handbook of Research Methods in Social and Personality Psychology*. Pending, 2014, ch. Behavioral observation and coding.

21. Jordan, B., and Henderson, A. Interaction analysis: Foundations and practice. *J Learn Sci 4*, 1 (1995), 39–103.

22. Kipp, M. ANVIL- a generic annotation tool for multimodal dialogue. *Eurospeech* (2001), 1367–1370.

23. Lasecki, W., Wesley, R., Nichols, J., Kulkarni, A., Allen, J., and Bigham, J. Chorus: A crowd-powered conversational assistant. In *UIST* (2013).

24. Lasecki, W. S., Murray, K. I., White, S., Miller, R. C., and Bigham, J. P. Real-time crowd control of existing interfaces. In *UIST* (2011), 23–32.

25. Lasecki, W. S., Song, Y. C., Kautz, H., and Bigham, J. P. Real-time crowd labeling for deployable activity recognition. In *CSCW* (2013).

26. Lasecki, W. S., Teevan, J., and Kamar, E. Information extraction and manipulation threats in crowd-powered systems. In *CSCW* (2014).

27. Lasecki, W. S., Thiha, P., Zhong, Y., Brady, E., and Bigham, J. P. Answering visual questions with conversational crowd assistants. In *ASSETS* (2013).

28. Lasecki, W. S., Weingard, L., Ferguson, G., and Bigham, J. P. Finding dependencies between actions using the crowd. In *SIGCHI* (2014), 3095–3098.

29. Lasecki, W. S., White, S., Murray, K. I., and Bigham, J. P. Crowd memory: Learning in the collective. In *CI* (2012).

30. Shamma, D. A., Kennedy, L., and Churchill, E. F. Tweet the debates: understanding community annotation of uncollected sources. In *SIGMM* (2009), 3–10.

31. Tiemens, R. K. Television's portrayal of the 1976 presidential debates: An analysis of visual content.

32. von Ahn, L. *Human Computation*. PhD thesis, Carnegie Mellon University, 2005.

33. Vondrick, C., Patterson, D., and Ramanan, D. Efficiently scaling up crowdsourced video annotation. *Int J of Comput Vision* (2012), 1–21.

34. Weingart, L. R., Olekalns, M., and Smith, P. L. Quantitative coding of negotiation behavior. *Int Negotiation 3*, 9 (2005), 441–456.

35. Zhang, H., Horvitz, E., Miller, R. C., and Parkes, D. C. Crowdsourcing general computation. In *SIGCHI Workshop on Human Compuation* (2011), 622–627.