Detecting low-quality crowdtesting workers

Ricky K. P. Mok, Weichao Li, and Rocky K. C. Chang
Department of Computing
The Hong Kong Polytechnic University
{cskpmok|csweicli|csrchang}@comp.polyu.edu.hk

***Abstract*—QoE crowdtesting is increasingly popular among researchers to conduct subjective assessments of different services. Experimenters can easily access to a huge pool of human subjects through crowdsourcing platforms. A fundamental problem threatening the integrity of crowdtesting is to detect cheating from the workers who work without any supervision. One of the approaches in classifying the quality of workers is analyzing their behavior during the experiments. A major challenge is to systematically analyze the mouse cursor trajectory. However, existing works usually analyze the trajectory coarsely, which cannot fully extract the information imbedded in the trajectory.**

In this paper, we propose to use finer-grained cursor trajectory analysis, including submovement analysis, to identify low quality workers. Our approach is to define a set of ten *worker behavior metrics* to quantify different types of worker behavior. A jQuery-based library was implemented to collect the worker behavior. Moreover, four different 5-point Likert scale rating methods were employed. A number of methods, including question design, instructions, and human inspections, are used to label workers into three categories. We then apply multiclass Naïve Bayes classifier to construct different models using all or some of the metrics and the workers' category. Our results show that the error rates of the model trained from four metrics is equal or less than 30% for four rating methods. By combining the predictions from the four rating methods, the successful rate in detecting low-quality workers is around 80%.

I. Introduction

QoE crowdtesting [1] is increasingly popular among researchers to carry out subjective assessments. Through crowdsourcing platforms (e.g., Amazon Mechanical Turk (MTurk) [2] or CrowdFlower [3]), they can evaluate the quality of experience (QoE) of different network services, such as video streaming [4], [5], VoIP, and IPTV [6]. Experimenters can easily deploy their experiments by compiling the assessments as a website published in the crowdsourcing platform. After finishing the assessments, the workers can report to the platform to claim their payment.

The advantages of using crowdtesting over traditional laboratory experiments are lower cost, and a larger and more diverse crowd of workers [1]. However, without any supervision, the quality of the works received from crowdtesting is questionable [7]. Some cheaters only intend to maximize their payment with minimum effort by quickly submitting the assessments. Even if workers may not intend to cheat, they can be distracted or incapable for the task. Both kinds of workers can result in unreliable measurements. Thus, identifying these workers can help improve the reliability of crowdtesting.

978-1-4673-7113-1/15/$31.00 ©2015 IEEE

Analyzing the worker behavior is a recent trend for inferring the quality of workers (e.g., [8], [9]). Worker behavior-based mechanism has three main advantages over existing anti-cheater methods, which will be reviewed in §II. First, as the monitoring of worker behavior runs at the background, it is almost invisible to the workers. Therefore, the low-quality workers who only focus on resolving the anti-cheating checks may not be alerted. Second, the time and cost of experiment can be reduced because the monitoring will not induce extra or redundant questions in the assessment. The third advantage is that our mechanism is independent of the assessment results, so the test items or stimulus are not required to have any implicit ranking or absolute answers.

Existing worker behavior based approaches focused on the timing of events, such as consolidation time or completion time. These metrics are useful, but we argue that the mouse cursor movement is also very important in measuring the quality of workers. Previous studies have shown that the mouse cursor movement can reveal the cognitive processes [10], [11]. These behaviors can provide implicit measures for the reliability of workers. However, it is challenging to systematically study the cursor trajectories.

In this paper, we propose novel worker behavior based metrics to classify the quality of workers. We particularly apply submovement analysis [12], which is common in the human-computer interface area to investigate the performance and accuracy of pointing devices, to process the cursor trajectories. Submovement analysis counts the micro-movements from the traces. In addition, we adopt part of the cursor measures proposed in [13] to quantify the cursor trajectories, such as the velocity and the acceleration of the cursor.

We evaluate our proposed metrics using worker behavior datasets collected from our adaptive video quality assessment crowdsourced through the MTurk and CrowdFlower. We firstly quantify the quality of workers by composing a *quality score* from multiple aspects of the assessment results, and then classify them into three groups according to their quality. We finally train a multiclass Naïve Bayes model [14] to classify workers using the metrics computed from workers' behavior. Furthermore, four rating methods suitable for rating Likert scale, are investigated, including radio buttons, stars, slide bar, and number steps.

Our results show that four out of ten metrics can effectively infer the workers' quality. The error rates of the trained model for all rating methods are around 30%. These metrics include the submovement count, the time delay and the cursor's speed, and the number of extra clicks. We also found that, among four rating methods, stars and radio buttons are more effective to use worker behavior based approach. By combining multiple rating methods, the accuracy of detecting low-quality workers can reach about 80%.

The outline of this paper is structured as follows. In §II, We first highlight some related works. We first elaborate the importance of worker behavior to this problem in §III. After describing our methodology in §IV, we present the experiment setup in §V and analyze the results in §VI. Finally, we conclude our paper in §VII.

II. Related Works

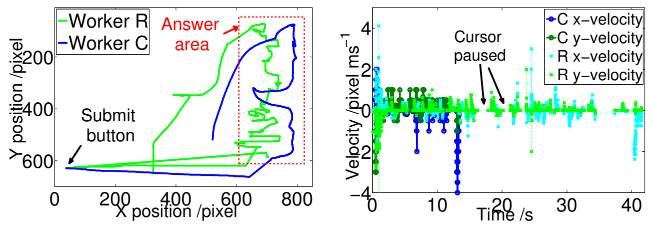
Some previous works employed worker behavior or application layer metrics to identify low-quality workers. Rzeszotarski et al. [8] proposed to use several user behavior to infer the quality of workers. However, they aggregated mouse cursor movements into events without storing the coordinates. Therefore, their analysis on mouse cursor movement was coarse. In their follow-up paper [15], the authors focused on visualizing the user behavior, which can help the experimenters to manually screen out potentially low-quality workers. Costagliola et al. [16] captured the student behavior in an e-learning system and detected cheating by analyzing the sequence of answering questions. Hirth at el. [9] analyzed some application layer metrics, such as consideration time and completion time, and then flagged the outliers as low-quality workers.

Other than analyzing worker behavior, some existing works proposed a better job design. For example, using CAPTCHAs or asking questions with known answers to evade software bots [17], and adopting a two-phrase approach to screen out a set of pseudo-reliable crowd before conducting the actual assessment [18]. Another kind of methods processes the data after the workers complete the tasks, such as comparing the data with the gold standard data [19], finding outlying workers by average and deviation of results [20], [21], and exploiting the ranks rated by workers [7].

III. Background

Estimating the quality of workers based on their behavior is one of the recent trends in crowdsourcing research. Several existing works discussed in §II mostly concern about the timing factors, such as the time between answering questions or the completion time. Although Rzeszotarski et al. [8] employ mouse movement as one of the metrics, they aggregate the movement as one event for moving every 200 px. Therefore, much useful information, including the direction of movement and the cursor's speed is lost.

We believe that these information hidden in the mouse cursor trajectory can also help infer the quality of workers, because cursor movements strongly correlate with the eye movement [22]. We manually select two workers from our experiment, an honest worker *R* and a low-quality worker *C*, to illustrate the importance of cursor trajectory. Fig. 1(a) shows the cursor trajectories of the two workers. The red dotted rectangle is the area for answering the questions. We can see that worker *C* takes a much direct pathway to answer all the questions, but worker *R* shows more zig-zag paths in between questions. Besides the pathway, the velocity of the cursor also carries useful information. Fig. 1(b) plots the velocity of the two traces in Fig. 1(a). We can see that worker *C* moves the cursor with much higher velocity and pauses with a shorter period between each movement than worker *R*.



(a) Cursor trajectories. (b) Cursor velocity.

Fig. 1. Worker *C*'s and Worker *R*'s cursor data.

However, it is challenging to systematically quantify the trajectory, because the trajectory can be affected by many factors, such as the responses and the worker's habit. In this paper, we apply submovement analysis [12] to capture the micro-structure in the trajectories. Fig. 2 shows an example of a cursor trajectory consisting of two submovements. The horizontal dotted line connects the start and end points. The first submovement is in upward direction away from the horizontal line. The second submovement changes to downward direction until reaching the end point. Furthermore, Hwang et al. [13] proposed a set of cursor measures, which are mainly based on submovement analysis, to analyze the performance and accuracy of pointing devices for different kinds of users. We also adopt the cursor measures to infer workers' quality.

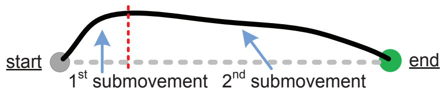


Fig. 2. A submovement example.

We compute the submovement from cursor trajectories using the following steps. We define function $\mathcal{S}(a, b)$ to be the number of submovements between time epoches a and b. The cursor trajectories we collected are represented by a series of points $\{t_i, x_i, y_i\}, \forall i = 0, 1, ..., n-1$, where x_i and y_i are respectively the x and y coordinates, and t_i is the timestamp of this datum. To compute $\mathcal{S}(t_0, t_{n-1})$, we compute the x and y components of the cursor velocity by $(x_{i+1} - x_i)/(t_{i+1} - t_i)$ and $(y_{i+1} - y_i)/(t_{i+1} - t_i)$, respectively. Finally we count the number of zero-crossings in either x or y velocity components as the number of submovement.

IV. Methodology

Our methodology is to first compute a set of *worker behavior metrics*, which is derived from the worker behavior collected when they answer the questions. We combine several existing methods to estimate the quality of workers. Then, we apply multiclass Naïve Bayes model to learn the relationships between the metrics and the quality of workers.

The worker behavior we collected includes cursor coordinates, mouse clicking/over/up-and-down positions, and mouse scrolling events etc. Furthermore, we install callback functions to each rating objects, such as radio buttons or text fields, to distinguish between random clicks and clicks on the rating objects. Other browser events, such as the sizing or loss of focus, are also recorded. Each record is timestamped at the user side with time resolution of 1 ms.

A. Worker behavior metrics

We compose ten worker behavior metrics from the behavior trace. This set of metrics extracts information from the start-up period, inter-question periods, and the overall time period. We believe that these three types of metrics can capture workers' cognitive process from different aspects. In the following sections, we will use the timeline in Fig. 3 as an example. The assessment page finishes rendering and starts capturing worker behavior at time t_0. The worker clicks on the i^{th} question at time $t_c^{(i)}$. The *start-up period* is defined as the time period from the page rendered to the first click on the answer, whereas the inter-question time periods are defined as the time period between the worker answering a question and the next one.

Each cursor movement record contains the coordinates, x_j and y_j, and its timestamp t_j, where j is the j^{th} cursor movement record in the trace. The shaded area is the time period in which continuous mouse cursor movement with inter-cursor movements less than 50 ms (i.e., $t_j - t_{j-1} < 50$ ms) is recorded. Otherwise, we treat the movement as a pause. We use $t_p^{(k)}$ and $t_p^{'(k)}$ to denote the beginning and the end of the k^{th} pause event, respectively. We let the total number of cursor movement records, clicks, and pauses to be N, C, and P, respectively. In the rest of the paper, we use these notations to introduce the computation of the worker behavior metrics.

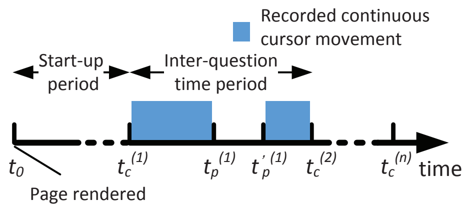


Fig. 3. A timeline of events.

1) Start-up time and submovement count: The workers may skim through the questions before answering them and moving the mouse cursor. We quantify this behavior by measuring the length and counting the submovement of the start-up period (i.e., $m_{st} = t_c^{(1)} - t_0$ and $m_{sc} = \mathcal{S}(t_0, t_c^{(1)})$, respectively).

2) Overall submovement count: We consider the total number of submovement throughout the task (i.e., $m_{tc} = \mathcal{S}(t_0, t_N)$). This metrics can quantify the micro-movement generated by the workers during the assessment.

3) Overall number of pause and median pause duration: Submovement can only reveal the direction of movement. To obtain the temporal measures, we consider the number of pause, P, and the median pause duration, m_{td}. Eqn. (1) shows the computation of m_{td}. We consider there is a pause event whenever the cursor stays at the same position for longer than 50 ms. We employ a shorter time than the one used in [8], because our task is relatively simple and the workers can answer quickly.

$$m_{td} = Md(\{t_p^{(i)} - t_p^{(i-1)} | i = 2, ..., P\}), \tag{1}$$

where $Md(\cdot)$ returns the median value of the input set.

4) Number of extra clicks: We count the number of *extra* clicks generated by the workers, denoted by m_{tk}. We subtract the minimum number of clicks required to complete the task from the number of clicks recorded by the trace. For example, we assume that only one click is required for answering a multiple choice question with radio buttons.

5) Median inter-question time and submovement: Besides the overall statistic of the page, we also consider the behavior during the inter-question period. We first slice the trace by the time the worker answering the questions. Then, we can compute the median length of time, m_{it}, and the number of submovement generated, m_{is}, in between answering each questions by Eqn. (2) and (3), respectively. These two metrics can quantify the timing and movement when the workers work on the task.

$$\begin{aligned} m_{it} &= Md(\{(t_c^{(i)} - t_c^{(i-1)}) | i = 2, 3, ..., C\}), & (2) \\ m_{is} &= Md(\{\mathcal{S}(t_c^{(i-1)}, t_c^{(i)}) | i = 2, 3, ..., C\}). & (3) \end{aligned}$$

6) Median cursor speed and acceleration: The median cursor speed, m_{cs}, and acceleration, m_{ca}, are the first and second derivatives of the coordinates which can be computed in Eqns. (4) and (5), respectively. These two metrics are important measures in characterising the cursor trajectories [23], [13].

$$\begin{aligned} \delta D(i) &= \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}, \\ \delta t(i) &= t_i - t_{i-1}, \\ m_{cs} &= Md(\{\frac{\delta D(i)}{\delta t(i)} | i = 2, 3, ..., N\}), & (4) \\ m_{ca} &= Md(\{\frac{\delta^2 D(i)}{\delta^2 t(i)} | i = 3, 4, ..., N\}). & (5) \end{aligned}$$

V. Evaluations

Our aim of the evaluations is to investigate the relationships between the worker behavior metrics and the quality of workers. In our evaluations, we publish crowdtesting tasks to evaluate the quality of experience (QoE) of different video bitrate adaptation schemes similar to [24]. Our crowdtesting task is implemented as a simple web site similar to [25]. Each worker was required to rate the QoE of four 60-second video clips. Our customized video player adjusts the video bitrate in three of the video clips according to a pre-defined scheme, while the remaining one, serving as a control, is kept at a constant highest/lowest bitrate. After playing each video, the worker was prompted to answer 16 questions.

A. Task Design

In addition to the single-item measure used in [26], we use 15 multiple choice questions and 1 open-ended questions to measure the QoE. The workers were asked to rate the QoE they just watched from various aspects, including picture/sound quality, video content, and the smoothness of the playback etc. The workers were required to indicate whether they noticed any video quality adaptation. Three reverse-coded questions are set to measure the worker's reliability. Besides, we have implemented four rating methods commonly used in rating 5-point Likert scales. One of the methods was randomly chosen for workers in each assessment.

The width of the question page is 800 px, which should be able to display on modern PCs without horizontal scrolling. Furthermore, we have implemented four common methods for rating 5-point Likert scale. They are radio buttons, slider bar, number field, and stars. We used the `RateIt` [27] jQuery library to implement the stars. The other three methods are the input types natively supported in HTML5. The size of all rating methods are unified as 200 px (width) $\times$ 20 px (height).

B. Capturing worker behavior

We have implemented a jQuery-based library to collect the worker behavior as described in §IV from the browser. The library is deployed in the question page, because answering the questions must involve multiple mouse movements and clicks. It starts recording right after the page is completely loaded. Our library is run at the background, and it returns the worker behavior to our server using AJAX every second. At the server side, we used `php` and `MySQL` to receive and store the data.

C. Estimating quality of workers

The subjective nature of QoE crowdtesting makes it infeasible to measure the accuracy of the workers. We tackle this problem by imbedding multiple cheater-detection tactics in our assessment, such that we can infer the worker's quality with some confidence. We compute a *quality score* to quantify the quality of workers. The score is composed of seven measures which are computed from the responses of all four video assessments and manual ratings. We assume the quality of worker does not change throughout the whole task. Therefore, we assign the quality score per worker instead of per assessment. The set of metrics can be classified into three types.

1) Complexness in text input response: There is an open-ended question in our assessment requiring the workers to input three words separated by commas about the content of the video she just watched. This question is similar to the image/video tagging tasks. Based on their answers, we can examine whether the worker had paid sufficient attention to the video and the question.

We analyze the response by three metrics (q_{wc}, q_{ww}, and q_{wf}), which refer to the number of unique characters used, the number of unique words used, and the format of the response. We also manually inspect the content of the response and give a rating to each, q_{ct}.

To compute q_{wc}, we first convert the response to small capital letters and then count the number of unique characters used. We normalise this index by dividing by 26, which is the total number of English alphabets. Another measure, q_{ww}, considers the ratio of unique words to the total number of words in the response. The unique words are found by grouping the same sub-string slitted by non-alphabet characters. We observe that some workers gave similar responses to all four videos with different content, such as "good" and "interesting". As the counting of characters or words cannot inspect the content of the responses, we also manually rate the responses (from 1 to 5) as a measure, q_{ct}. Because our tasks only require the workers to input three words per assessment, the rating criteria mainly focus on the *accuracy* of the responses rather than the descriptiveness. The scores are normalised by dividing by 5.

As the counting of characters or words cannot inspect the content of the responses, we also manually rate the responses (from 1 to 5) as a measure, q_{ct}. Because our tasks only require the workers to input three words per assessment, the rating criteria mainly focus on the *accuracy* of the responses rather than the descriptiveness. The scores are then normalized by dividing 5.

2) Violation of soft rules: To ensure the workers reach a certain quality, we have some rules stated in the instructions. For example, the workers have to watch the whole video without fast forwarding. Violating these "hard" rules can lead to a rejection of his work or stopping him from proceeding to the next assessment. On the other hand, "soft" rules do not led to rejection, but they can reflect the consciousness of the workers. We have included two soft rules in our assessment. One of them is implemented in a question requiring the workers to indicate whether they notice any video quality adaptation. The workers were instructed to skip the next question if they did not notice any quality changes. However, we find that some workers did not skip the question as instructed. q_{jp} is the average number of correctly followed the rules across the four assessments in the whole task.

Another soft rule is about the formatting of the text responses, which requires the workers to input three comma separated words. Although it is feasible to enforce the formatting policy at the browser before the workers submit the answer, we do not limit the input. Therefore, we can capture the low-quality workers who input casually. We find that around 18% of workers did not input the correct format in all four assessments. Similar to q_{jp}, this measure, q_{wf}, is computed using the average number of correctly formatted input.

3) Contradictory responses: Low-quality workers tend to provide random ratings or the same rating for all questions. These workers can be easily screened out by applying reverse-coded questions. These questions are phrased in the semantically opposite direction to another one. For example, "The initial picture quality is too low." vs. "The initial picture quality meets my expectation." In our assessments, three questions are reverse-coded. We compose a measure, q_{rc}, which computes the average differences in ratings between the positively and negatively coded questions in all four assessments as shown in Eqn. (6).

$$q_{rc} = 1 - \frac{\sum_{j=1}^{4}\sum_{k=1}^{3} \Delta r_j k}{3 \times 4 \times 5}, \tag{6}$$

where Δr_k is the difference in ratings of the k^{th} in the j^{th} assessment.

The last measure, q_{cn}, checks whether the workers can correctly identify the video bitrate adaptation throughout the video streaming. Because it is easy for workers to determine whether the video quality has changed or kept constant, we believe that this measure can reveal the level of concentration to the assessment. This measure averages the number of correctly identified assessments in the task.

Finally, the quality score, q, is the summation of the seven measures. Therefore, the score is between 0.4 to 7. A higher score means a better worker's quality.

VI. Results

We published our crowdtesting task to two major crowdsourcing platforms, Amazon Mechanical Turk (MTurk) and CrowdFlower. Only US workers were able to perform our tasks of evaluating the video steaming performance. We chose the workers with acceptance rate higher than 90% in MTurk and the "highest quality" in all available channels in CrowdFlower. Each completed worker was awarded \$0.5 US.

We successfully collected results from 172 workers (42 MTurk workers and 130 CrowdFlower workers from 20 channels). As each worker was required to rate four video clips, we collected a total of 688 (=172 $\times$ 4) samples of worker behavior data. The median time for {MTurk/CrowdFlower} workers to complete one assessment is {67.8s/71.3s}, respectively. About 90% of the workers submitted the assessment in 150s, but we also found a few workers taking longer than 5 minutes. Fig. 4 plots the CDF of the quality score of all workers. Besides, we also plot the CDFs of the quality score from MTurk and CrowdFlower workers separately. We can see that workers from MTurk generally have a lower score than those from CrowdFlower. The median score for {MTurk,CrowdFlower} workers is {4.66,5.22}.

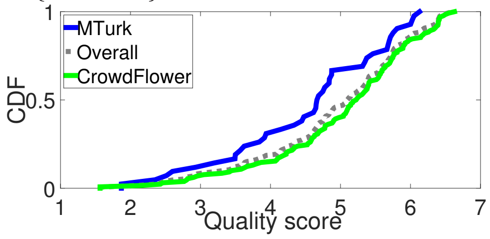


Fig. 4. The CDFs of the quality scores.

A. Correlating the worker behavior metrics with the quality score

We apply a multiclass Naïve Bayes classifier to infer workers' quality from the worker behavior metrics. To simplify the model, we categorise the workers into three groups according to their quality scores. A worker with a quality score {less than 3/between 3 and 4.5/greater than 4.5} is labeled as {low-quality/marginal/acceptable} worker, because the quality score {less than 3/less than 4.5} means the worker can only meet less than {half/two-third} of the seven measures. In our dataset, around {8%/20%} of workers are labeled as {low-quality/marginal}. We also analyze different rating methods individually as we believe that they could trigger different mouse cursor movement.

To train the predictive model, we include all or some of the worker behavior metrics and compare their error rates for the best model. Some metrics are computed as count data. We apply multinomial distribution to these variables, which are denoted by ‡ in Table I. In the following, we adopt a 10-fold cross validation method to estimate the error rate of the classifiers. Because the Naïve Bayes classifier is a probabilistic classifier, the error rate of each model is the average error rate computed from 100 times of training.

We first apply all the ten worker behavior metrics into the model as shown in the first column of Table I. Fig. 5 plots the average error rates of the model with 95% confidence intervals. In model 1, stars and slide bar rating methods can achieve error rates of 31.8% and 33.1%, respectively. However, the error rates for radio buttons and number field are large (i.e., 38.3% and 59.0%, respectively).

We find that the poor results are resulted from some metrics which cannot fit the model well. To select the metrics to be included in the model, we use a simple linear regression model to identify the metrics which can better predict the quality score. It is found that four of the metrics are statistically significant in at least one of the rating methods: m_{sc}, m_{it}, m_{cs}, and m_{tk}. Therefore, we select these four metrics to form models 2 to 5 and re-train the model. Subsequently, the average error rates are decreased to around 30%. Among the four rating methods, the radio button and stars are better other two methods with an error of 27.2% in model 5. All rating methods except the number field method also obtain the minimum error rates in model 5, whereas the number field method achieves the lowest error rate in model 3. Therefore, we suggest that different sets of worker behavior metrics should be used when the rating method is different.

TABLE I
Worker behavior metrics used in various models.

Model	Worker behavior metrics									
	m_{st}	m_{sc}‡	m_{tc}‡	P‡	m_{td}	m_{it}	m_{is}‡	m_{cs}	m_{ca}	m_{tk}‡
1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2		✓				✓		✓		✓
3		✓						✓		✓
4		✓				✓		✓		
5		✓				✓				✓

Note: ‡the variables used multinomial distribution.

We further analyze the accuracy of the models. For each rating method, we use the model with the lowest error rate to predict the worker's category. We then compute the differences between the actual and the predicted worker's category (i.e., $\Delta L = l - \check{l}$, where l is the actual category, and $\check{l}$ is the predicted category). Table II shows the percentage of workers in each category. The accuracy for all four rating methods ($\Delta L = 0$) is

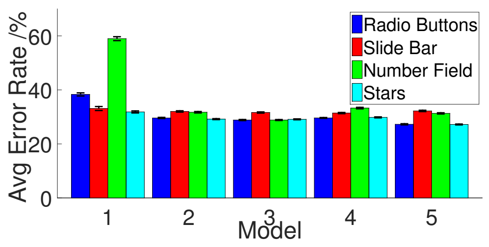


Fig. 5. The average error rate of different models.

quite high, from 73.3% to 82.6%. The quality of less than 2% of workers is under-estimated by two categories ($\Delta L = 2$). This can cause false alarm that the experiment wrongly labels the worker as a cheater. However, it is still acceptable in practice, because the honest workers can complaint to the experimenter on his decision. The experimenter can also manually check those workers labeled as low-quality. On the other hand, there are around 16% and 6% cases over-estimate by one and two categories, respectively. The false negative rate is barely satisfactory.

However, we find that the predicted workers' quality could be different among the four rating methods. Thus, we decide the workers' quality as the lowest category estimated by all rating methods. The fifth row in Table II shows the combined results. The accuracy can be retained, whereas the false negative rate ($\Delta L = -2$) is significantly reduced to 0.58%.

TABLE II
THE PERCENTAGE OF WORKERS SHOWING DIFFERENCES BETWEEN THE ACTUAL AND PREDICTED WORKER'S CATEGORY.

Rating methods	ΔL				
	-2	-1	0	1	2
Radio buttons	4.07%	12.8%	82.6%	0.58%	0%
Slide bar	5.23%	17.4%	75.6%	1.74%	0%
Number field	4.65%	15.1%	73.8%	4.65%	1.74%
Stars	6.98%	16.3%	73.3%	2.91%	0.58%
Combined	0.58%	8.72%	82.6%	6.40%	1.74%

VII. Conclusion

This paper presented a novel worker-behavior based metrics which could be used to infer the quality of workers. We proposed to extract information from the cursor trajectory and quantify them by using a set of worker behavior metrics. In our experiment, we carefully designed a crowdtesting task, such that we could estimate the quality of workers with a quality score. Four different rating methods were examined. We then correlated the worker behavior metrics with the score using multiclass Naïve Bayes model. Our results showed that the error rate for the models with three metrics is less than 30%. We also found that different sets of metrics should be used for different rating methods. By combining the predictions from the four rating methods, the successful rate in detecting low-quality workers is around 80%.

Acknowledgements: We thank three anonymous reviewers for their valuable comments. This work is partially supported by an ITSP Tier-2 project grant (ref. no. GHP/027/11) from the Innovation Technology Fund in Hong Kong.

References

[1] Tobias Hoßfeld, Christian Keimel, Matthias Hirth, Bruno Gardlo, Julian Habigt, Klaus Diepold, and Phuoc Tran-Gia, "Best practices for QoE crowdtesting: QoE assessment with crowdsourcing," *IEEE Trans. in Multimedia*, vol. 16, no. 2, pp. 541–558, 2014.
[2] Amazon, "Mechanical turk," https://www.mturk.com.
[3] "Crowdflower," http://www.crowdflower.com/.
[4] Christian Keimel, Julian Habigt, and Clemens Horch, "Video quality evaluation in the cloud," in *Proc. IEEE PV*, 2012.
[5] Óscar Figuerola Salas, Velibor Adzic, Akash Shah, and Hari Kalva, "Assessing internet video quality using crowdsourcing," in *Proc. ACM CrowdMM*, 2013.
[6] Kuan-Ta Chen, Chi-Jui Chang, Chen-Chi Wu, Yu-Chun Chang, and Chin-Laung Lei, "Quadrant of euphoria: a crowdsourcing platform for QoE assessment," *IEEE Network*, vol. 24, no. 2, pp. 28–35, 2010.
[7] Chen-Chi Wu, Kuan-Ta Chen, Yu-Chun Chang, and Chin-Laung Lei, "Crowdsourcing multimedia QoE evaluation: A trusted framework," *IEEE Trans. in Multimedia*, vol. 15, no. 5, pp. 1121–1137, 2013.
[8] Jeffrey M. Rzeszotarski and Aniket Kittur, "Instrumenting the crowd: Using implicit behavioral measures to predict task performance," in *Proc. UIST*, 2011.
[9] Matthias Hirth, Sven Scheuring, Tobias Hoßfeld, Christian Schwartz, and Phuoc Tran-Gia, "Predicting result quality in crowdsourcing using application layer monitoring," in *Proc. IEEE ICCE*, 2014.
[10] Joo-Hyun Song and Ken Nakayama, "Hidden cognitive states revealed in choice reaching tasks," *Trends in Cognitive Sciences*, vol. 13, no. 8, pp. 360–366, 2009.
[11] Jonathan B. Freeman and Nalini Ambady, "Mousetracker: Software for studying real-time mental processing using a computer mouse-tracking method," *Behavior Research Methods*, vol. 42, no. 1, pp. 226–241, 2010.
[12] David E Meyer, Richard A. Abrams, Sylvan Kornblum, Charles E. Wright, and J. E.. Keith Smith, "Optimality in human motor performance: Ideal control of rapid aimed movements," *Psychological Review*, vol. 95, no. 3, pp. 340–370, 1988.
[13] F Hwang, S Keates, P Langdon, and J Clarkson, "A submovement analysis of cursor trajectories," *Behaviour & Information Technology*, vol. 24, no. 3, pp. 205–217, 2005.
[14] Jason Rennie, Lawrence Shih, Jaime Teevan, and David Karger, "Tackling the poor assumptions of naive bayes text classifiers," in *Proc. IMLS ICML*, 2003.
[15] Jeffrey M. Rzeszotarski and Aniket Kittur, "Crowdscape: Interactively visualizing user behavior and output," in *Proc. UIST*, 2012.
[16] Gennaro Costagliola, Vittorio Fuccella, Massimiliano Giordano, and Giuseppe Polese, "Logging and visualization of learner behaviour in web-based e-testing," in *Proc. ICWL*, 2007.
[17] Julie S. Downs, Mandy B. Holbrook, Steve Sheng, and Lorrie Faith Cranor, "Are your participants gaming the system?: screening mechanical turk workers," in *Proc. ACM CHI*, 2010.
[18] Mohammad Soleymani and Martha Larson, "Crowdsourcing for affective annotation of video: Development of a viewer-reported boredom corpus," in *Proc. ACM CSE*, 2010.
[19] Sabine Buchholz and Javier Latorre, "Crowdsourcing preference tests, and how to detect cheating," in *Proc. ISCA INTERSPEECH*, 2011.
[20] Flávio Ribeiro, Dinei Florêncio, Cha Zhang, and Michael Seltzer, "Crowdmos: An approach for crowdsourcing mean opinion score studies," in *Proc. IEEE ICASSP*, 2011.
[21] Flávio Ribeiro, Dinei Florencio, and Vítor Nascimento, "Crowdsourcing subjective image quality evaluation," in *Proc. IEEE ICIP*, 2011.
[22] Mon Chu Chen, John R. Anderson, and Myeong Ho Sohn, "What can a mouse cursor tell us more?: correlation of eye/mouse movements on web browsing," in *Proc. ACM CHI*, 2001.
[23] James G. Phillips and Thomas J. Triggs, "Characteristics of cursor trajectories controlled by the computer mouse," *Ergonomics*, vol. 44, no. 5, pp. 527–536, 2001.
[24] Ricky Mok, Xiapu Luo, Edmond Chan, and Rocky Chang, "QDASH: A QoE-aware DASH system," in *Proc. ACM MMSys*, 2012.
[25] Benjamin Rainer, Markus Waltl, and Christian Timmerer, "A web based subjective evaluation platform," in *Proc. QoMEX*, 2013.
[26] ITU-T Recommendation P.911, *Subjective audiovisual quality assessment methods for multimedia applications*, December 1998.
[27] "RateIt plugin for jQuery," http://rateit.codeplex.com/.