It is about Time: Time Aware Quality Management for Interactive Systems with Humans in the Loop

Markus Krause  
University of Hannover  
Research Group HCI  
Hannover, Germany  
markus.krause(at)hci.uni-hannover.de  

Robert Porzel  
University of Bremen  
Research Group Digital Media  
Bremen, Germany  
Porzel(at)tzi.de

Abstract  
In recent years crowd-based and human computation systems have attracted increasing attention in science and industry. For applications that are driven by input from a multitude of human raters, ensuring data reliability and organizing an interactive workflow constitute a new challenge. In this paper we describe a novel approach to ensure data quality in crowd-based and human computation systems. The proposed algorithm features the potential for direct feedback and interactivity while producing little computational overhead.

Author Keywords  
Human computation; Crowd-sourcing; Interactive Systems; Quality Management

ACM Classification Keywords  
H.1.2 User/Machine Systems:  
Human Factors;  
Human information processing

General Terms  
Human Factors; Algorithms; Measurement

Introduction  
Computational systems with human contributors in the loop bear great promises but also have to overcome substantial challenges. A major one is to ensure data
quality. Responses collected from a human contributor are hard to evaluate with purely algorithmic methods. Tasks, such as image or audio categorization, need a human judge to determine answer quality. This paper proposes a new algorithm that estimates contributor quality for each individual response. It assesses the quality of one response by analyzing previously given ones. The proposed system estimates the quality of a response using an adapted kappa coefficient. This statistic measurement predicts the quality of an answer using a fixed sized sliding window and allows for bias correction. The algorithm performs in constant time and with minimal resource locking. It therefore qualifies as a method for highly interactive scenarios and is implemented as a restful web service which allows for iterative integration of contributor responses.

State of the Art
Initial solutions to address evaluation in crowd systems relied on redundancy and over-classification. Therein, multiple contributors complete the same task. With methods, as majority voting, it is consequently feasible to identify the correct answers with acceptable likelihoods. One drawback of this approach is that redundancy is costly: if 10 contributors complete the same task, the cost of the crowd-sourced solution is comparable to an in-house one. A more efficient solution to this challenge is a modified Expectation Maximization algorithm [3]. There over-classification is used to identify correct answers and to measure contributor response quality. This approach also separates unrecov-
erable error rates from bias.

Other approaches provide assessment methods for crowd scenarios [6,10,11] that yield good results in many situations but come with shortcomings in highly interactive scenarios, such as human computation games, are not suited for real time systems [1]. One almost real time method exists [4] albeit it still requires costly computations. In contrast our approach performs bias correction and has lower costs in terms of time. Especially in gaming contexts we found that numerous accurate answers were lost as the established algorithms tend to average the contributors response quality over time. A solution in a gaming scenario is to use input or output agreement [8]. This effective method limits the game design possibilities significantly [7].

Time Aware Response Evaluation
One of the central aspects of usability in interactive systems is concerned with utility – a coin that has two sides in crowd sourcing and human computation systems: the primary utility of the system for the users as well as the reverse utility for the owners of the system whose primary concern is to obtain as much high quality data as possible. The core idea of our proposed algorithm is to estimate the quality by looking at the change of contributor behavior over time. The algorithm estimates the quality of each response of a user separately instead of calculating an average score over all seen responses. To calculate the quality of a contributor one can calculate the agreement between responses from the user and the system’s model. This is only possible if a certain initial knowledge is available to the system. To estimate the quality of a response some methods use a gold standard. This gold standard consists of a partial set of requests and expected optimal responses. This gold standard initiates the system’s model, which allows for estimating contributor response quality without aggregated knowledge on the whole task. Our algorithm tracks response quality for all responses of a contributor individually. Therefore, the algorithm needs gold standard requests to be presented to the user throughout his or her work on the task.

As a gold standard provides an initial model of the task the quality of a contributor can be estimated comparing responses from the user to the current model of the
This can be done building a confusion matrix [9] based on the responses of the user. As a measurement for the quality of a user we calculate the agreement between contributor and system based on the confusion matrix. A method to calculate such agreement between two raters is Cohen’s kappa [2]. Cohen’s kappa calculates the agreement $k$ with $P(o)$ expressing the observed agreement of both raters and $P(e)$ the expected agreement due to chance. Equation (1) calculates $P(e)$ and $P(o)$ for user $w$. With $c$ number of possible different responses, $g$ number of responses of user $w$, and $m^w$ the confusion matrix for user $w$.

\[
    k = \frac{P(o) - P(e)}{1 - P(e)} \tag{1}
\]

\[
    P(o) = \frac{1}{g} \cdot \sum_{i=1}^{c} m^w_{ii} \tag{2}
\]

\[
    P(e) = \frac{1}{g^2} \cdot \sum_{i=1}^{c} \left( \sum_{j=1}^{c} m^w_{ij} \cdot \sum_{k=1}^{c} m^w_{ki} \right) \tag{3}
\]

With the given equations (1-3) all responses would have the same kappa value, as it averages the quality of the contributor. Changes in user response quality could not be detected. To calculate agreement and estimate the quality of a response for a certain range around this response our algorithm uses a sliding window. Only responses within this sliding window are used to calculate kappa values. This way each response receives a kappa value based only on the surrounding responses. This sliding window selects a set of responses $R_{wnt}$ from the set of all responses $R_w$ from user $w$. Where $i$ is the index of any response from user $w$ and $n$ the index of the response the kappa value shall be calculated for. The sliding window selects up to $l$ responses before and $l$ after $n$ as seen in equation (4).

\[
    R_{wnt} = \{ r_i \in R_w | n - l \leq i \leq n + l \} \tag{4}
\]

As the system’s model would not be accurate for requests that are not in the gold standard, the proposed algorithm cannot use fixed observations. It needs to accumulate probability distributions. For each request the system keeps a probability distribution. Each response of $R_{wnt}$ is associated to exactly one request and the probability distribution for this request. Instead of adding a single discrete observation the system adds the probability distribution to the row vector of the confusion matrix. Not all responses contradicting the current model of the system originate from undesired or bad behavior of the contributors. A contributor can for instance be biased or have misunderstood the task which results in honest but invalid responses to requests.

With the confusion matrix it is possible to calculate a corrected answer for such cases. For every response from a user the probability distribution is given by the row vector of the confusion matrix of the user. The probability that user $w$ responds with $R_i$ for a request actually being $R_j$ given the response $R_{wnt}$ is calculated as seen in equation (5) as has been proposed before [5].

\[
    P_{wn}(R_j) = \frac{1}{g} m^w_{wnj} \tag{5}
\]

As the algorithm performs the proposed bias correction kappa from equation (1) is no longer suited to calculate the quality of a user’s response. Kappa estimates the agreement between two raters. With the bias correction in place the agreement is no longer important as long
as the disagreement is constant. Instead the proposed algorithm uses the best fit kappa to calculate contributor response quality. Best fit kappa $k_i$ estimates $P(e)$ and $P(o)$. To estimate the correct response the system aggregates all responses to that request.

**Evaluation**

The proposed algorithm has one important parameter $l$. This parameter determines how many responses are used to calculate the confusion matrix. The optimal value of $l$ depends on the distribution of gold standard requests in the data set. To estimate the relation between gold distribution and $l$ we used a synthetic data set. This set syn100 contained 100 requests for which 2 responses (true, false) were possible. It contained 20 users each giving 100 responses so that each request aggregates 20 responses. The initial over classification rate of this data set is 20. The set had a ratio of 0.25 requests for response “true” and 0.75 “false”. For large data sets the distribution of possible responses is mostly unknown. As most algorithms our proposed one sets its initial distribution to a normal distribution if there is no assumption on the real distribution. In the case of two possible responses this would be 0.5 for “true” and 0.5 for “false”. Therefore the algorithm has to adapt from its initial expected response distribution to the distribution in the data set. Even though the distribution can be controlled with delivering gold standard requests ratios around 0.25:0.75 are feasible. The quality of the responses of each user changes over time. The first 25 responses of the user are valid. The following 75 responses were randomly chosen to be either “true” or “false” ignoring the true response of the request. First we estimated the necessary ratio of gold requests necessary to reliable estimate contributor response quality. The measurement used was Cohen’s Kappa, this is not be confused with the Kappa calculations from the proposed algorithm. The final estimated responses for each request from the algorithm were compared to the expected correct responses. There were five different gold ratios tested (0.05, 0.10, 0.20, 0.33, 0.50). Each condition was repeated 100 times to calculate standard deviation. The system generated a random set and a random permutation of contributors from the 20 available in syn100 for every repetition. The over classification of each generated data set was 15. Figure 2 shows the results of this experiment. Good results could already be achieved with a gold ratio of 0.2. The measured Cohen’s Kappa value mean is 0.937 with a standard deviation of 0.07. Error rates for response “true” are higher than for “false”. Mean error rates are 0.081 SD (0.04) for “true” and 0.003 SD (0.01) for “false”. Based on these results the next experiment estimates an adequate value for $l$. The experiment again uses syn100.

![Figure 1: Cohen's Kappa values giving the agreement between our algorithms prediction and the correct results. The results are calculated for different values of parameter $l$.](image)
We measured the quality of the algorithm for all values \( l \) in the range 1 to 40. The system generated a random set and a random permutation of contributors from the 20 available in \textit{syn100} for every repetition. The over classification of each generated data set was 12.5. The quality of each case was measured as in the first experiment. As seen in Figure 1 the quality of the algorithms estimation rises quickly. However values for \( l \) that are more than the length of the high quality interval of the contributor reduce the estimation quality of the algorithm. This is plausible as the algorithm starts to average over correct and random responses.

The experiments show that good results can be obtained for data sets with a gold ratio of 0.2 and \( l \) values between 10 and 20. However to achieve high quality results the intervals of the contributors in which they give good results have to be at least the same size as the range \( l \) to be detected reliably. To finally evaluate the quality of the algorithm using the estimated parameters three different test sets were used. The first test set was again a synthetic set \textit{syn2500}. The set contains 2500 requests for which 2 responses (true, false) were possible. It contains 232 contributors each giving 100 responses so that the initial over classification rate is approximately 9. Similar to \textit{syn100} the first 25 responses of a contributor are valid. The following 75 responses were randomly chosen to be either “true” or “false” ignoring the true response of the request. The data set had a ratio of 0.25 requests for response “true” and 0.75 “false”. The second and third data set were acquired from 12 real contributors of which half were asked to contribute honestly and half to cheat. All contributors gave 2115 responses to 190 requests. The task was to categorize images into 4 categories, multiple assignments were possible. The algorithm estimated whether a category was assigned correctly or not.

The 190 images of the test set were chosen from a set of hand labeled images. Three raters previously labeled the initial set and only images with total agreement were chosen for the test set. From this two sets were derived \textit{Random} and \textit{Sorted}. In the \textit{Random} set responses were not ordered in the way they were given but were evaluated in a random order. All three test sets were prepared with a gold ratio of 0.2 and evaluated using: majority vote (MV), single best contributor (SB), the Dawid and Skene approach from Ipeirotis (IP), and our proposed algorithm (DK 15). For IP we used standard parameters as proposed here [5]. Our algorithm was used with a range value of \( l = 15 \). Figure 2 shows the results of this experiment. The results show that in scenarios were a lot of noise is to be expected majority vote (MV) does not perform well.

![Figure 2: Cohen’s Kappa values giving the agreement between an algorithms prediction and the correct results for three different test sets.](image_url)
Using only the answers of the best contributor (SB) is applicable when enough contributors are available. In cases were only few contributors are expected to respond constantly reliable only statistical methods such as IP work reliable. However in situations where noise appears in intervals the proposed algorithm performs best. The algorithm also allows for interactive use cases such as games or social network applications.

**Conclusion**

In this paper we have introduced and evaluated an approach that addresses the direct evaluation of user contributions. The issues tackled encompass the reverse utility of the corresponding crowd sourcing system by providing a running estimate of the data quality obtained by the respective system. This yields the potential for direct feedback of the ensuing utility measurements to the individual contributors. We have shown that our approach is both accurate and scalable at minimal computational cost given the existence of a small scale gold standard of pre-annotated data.

**REFERENCES**


