Reducing Cost & Increasing Profit in Microwork™: A Smart Gating Algorithm

Laura Gerhardt & Chaya J. Nayak
Goldman School of Public Policy

Introduction:

Samasource is a nonprofit organization based in San Francisco, CA that brings digital work to women and youth in poverty using a Mircowork™ model. Samasource's Microwork™ model breaks down digital projects into small tasks, which are then sent to individual workers through the internet using the organization's Samhub online Platform.

The Samahub platform uses a quality assurance algorithm to track worker performance on tasks and to ensure that overall project quality remains above a given threshold set by organization's contract with a client. The SamaHub is able to track worker performance through use of "gold" tasks or tasks that have verified answers. Gold, which is distributed in 20 question "qualification bursts" ensures that worker quality remains above a percentile threshold set by the client, by comparing a worker's average quality on gold tasks with the given threshold. If a worker's quality is below a given threshold, the worker is "gated," or removed from the project until they are trained by a manager. Currently, the gating process works by distributing 20 gold tasks for every 200 project tasks a worker completes.

Section 1 of this report further discusses the problems with Samasource's current gating algorithm with respect to cost and profit on a project. This report presents a solution to the problems apparent in Samasource's current gating algorithm: A Smart Gating Algorithm. In Section 2, we summarize other publications that influenced the development of our algorithm with regard to maintaining accuracy. As detailed in Section 2, this algorithm categorizes workers by performance, and thereby increases or decreases the number of tasks a worker must complete prior to working on a gold batch. Finally, we conclude by discussing the next steps that Samasource, and other organizations can take, after implementing the Smart Gating Algorithm, in order to increase project accuracy while decreasing cost relative to quality assurance.

Problem:

Every worker, regardless of their performance above or below the threshold is treated the same by the Smart Gating Algorithm. Although this algorithm is effective in ensuring that the quality of a project remains above a set quality threshold, it is a costly quality assurance measure for Samasource because it does not minimize the number of gold tasks workers complete.

The current gating threshold is unable to react to unique workers, thus distributing the same number of questions to a high quality worker who performs well above a given quality threshold, as a low quality worker who performs below a quality threshold. In treating all workers the same, Samasource is using too much gold on high quality workers, and too little gold on low quality workers, and thus is unable to maximize the impact of gold tasks, which are resource intensive to produce and may not be reused because of concerns with recognition of gold answers by workers which may decrease the confidence in quality estimation. Samahub also sometimes runs out of gold to distribute to workers, causing the Samahub to forgo qualification bursts. If gold runs out when there is no oversight of the hub, workers with low quality scores may compromise large portions of project tasks thus significantly decreasing accuracy of entire project batches. Figure 1 and 2 demonstrates the current gold gating algorithm and the new Smart Gating Algorithm.

Our Model:

To create the Smart Gating Algorithm, we relied upon data from completed tasks for one Samasource client, Client "A." Samasource collects and maintains a data warehouse of all prior projects completed by the organization's workers over the past three years. From the data warehouse, we extracted a dataset of all projects completed for Client "A" that included the following parameters:

1. **User ID:** A unique ID associated with a Samasource worker.
2. **Batch ID**: A unique batch ID associated with a given project for Client "A".
3. **Delivery Center ID**: There are several Samasource Delivery Centers within which workers are employed. Each Delivery center is associated with a unique categorical ID.
4. **Gold Metric Duration:** The time a took in seconds for a user to complete a task
5. **Gold Metric Correct:** A binary indicator that indicates whether a worker answered a given gold task correctly or incorrectly.

After dividing our data warehouse extraction into two training sets and one cross-validation set. Our first data processing task was to extract from the individual task records, Batch ID averages for both accuracy as well as the time taken on the project gold tasks (Gold Metric Duration). We expected a lot of variability in performance across batches, so we normalized each task accuracy and time score for a given batch into a z-score relative to the other users that worked on the batch (Figure 3). We recognize that this approach favors users that perform few tasks. To gauge a broader sense of a user's' overall performance, we then averaged her normalized performance time and accuracy scores. While metrics are relative to the other users on the project, we felt we had to account for the relative difficulty of the batch even if it meant it may compromise the independence of our estimates.

We subsequently normalized our averages we used k-means to establish clusters of time and user performance. Given the variability in the resulting centroid location, we considered inputting an initial centroid location, but ultimately implemented the traditional k-means algorithm in which initial centroid values are randomized (Figure 4). Though this variability is not ideal, we believe this is preferred to using support vector machines, because we want three classes of user performance: good, bad, and average. After implementing k-means, we used the resulting centroids as threshold cutoffs for determining good, bad, and average performance in out gating algorithm.

These thresholds were then weighted by the regressors of the of a multivariate regression of current accuracy score (not-normalized) on time and performance. In our regression model, we initially controlled for tenure and delivery center IDs. We dropped these variables from our regression model, because tenure did not add any explanatory power, and the delivery centers were collinear with the other variables. The resulting model regressed past score and time, to produce the following weights (Figure 5):

Time Weight: 0.0143776047882
Past Score Weight: 0.00519122132925
(**R-Squared** = -0.98963)

To validate this regression, we collapsed the accuracy scores into a categorical variable of good, bad, and average current perform-

ance. We then performed Naive Bayes on this performance category variable with the past performance and time variable and were able to predict the correct user type 98.8% of the time.

These weighted differences were then multiplied by the initial base question amount parameter (200) to increase or decrease the number of production tasks before gold. We then adjusted their score based on their current task performance. In modeling 6 different adjustment algorithms. We believe that our threshold modeling is the most effective. We then ran these models against our training set and our cross-validation set and learned that our naive bayes performed even better, with an accuracy of 99% and an R-squared value of 0.98, though our regression coefficients were higher.

Results:

From our model we found that user's performed an average of 41 more tasks before the gold tasks with the 2 centroid model and 147 more tasks before gold task with the 3 centroid model over an entire batch. Due to the increase in production tasks before gold, we see a decrease of 731 gold in terms of gold tasks needed for workers to complete a project. We estimate cost savings $20 per project due to the $0.03 cost of creating gold. Due to the reduction of creating gold tasks, with an average of 8 projects a month, Samasource could save approximately $1920 a year.

Samasource may also increase profit as workers will be able to spend more time working on production tasks, instead of gold tasks. The profit function estimates the profit Samasource will make on a given batch set. A batch set accounts for all of the tasks a worker completes before they are distributed a gold set. In pricing project contracts for clients, Samasource has developed cost and profit indicators that we used to steer our analysis. For Client "A," Samasource is paid $0.0348 per task completed. Thus the revenue from a task can be calculated by $0.0348q, where q is questions completed before gold by a worker. The cost function can then be calculated relative to the time a worker takes to complete a task multiplied by worker pay, plus the time a task spends in review multiplied by the percentage of all tasks reviewed and review pay, and finally the time a task spends in administrative or headquarter review multiplied by the percentage of all tasks reviewed and headquarter review pay. Since the "Smart Gating Function" increases overall quality of work by increasing gating for poor workers, and decreasing gating for good workers, the cost function for the current gold algorithm demonstrates higher percentages of tasks that must undergo review, and thus increased costs.

The resulting profit functions are as follows:

Profit for Smart Gating Algorithm = (0.0348q)-(.0008q + .000225q+.0000066q)
Profit for traditional Gating Algorithm = (0.0348q)-(.0008q + .00045q + .0000132q)

Through the profit function, we are not only able to verify that the Smart Gating Mechanism saves Samasource money, but also that the 3 centroid model produces the largest cost savings for the organization. With this profit function, we estimate that Samasource could save approximately $229 per project by implementing our algorithm with 3 centroids while maintaining accuracy. The cost savings of $229 is significant in that Samasource does 6-10 projects with Client "A" in a one month period. Over a year, the Smart Gating Algorithm can result in an average of $21,984 in savings over a year.

Related Work:

Other research has examined similar crowdsourcing platforms. Mason & Watts (2009) examined the financial incentives present in the crowdsourcing platform run by Amazon (Mechanical Turk). They found that there were no improvements in accuracy driven by financial reward, but speed was improved. The Samasource model, however, does not pay workers by task but rather adheres to bulk staff pricing. From Kumar and Lease, we learn that the accuracy of single labeled data can be poor when there is a wide variance for classifiers (Kumar & Lease 2011). This reiterates our need to evaluate individual worker performance throughout their crowdsourcing activities.

Next Steps:

We built a function that would then take the resulting output of our questions before gold and save it for the next qualification bursts and then run the same algorithms, while adjusting the weight on current performance to be the regression of the change in performance on number of questions. This function will be very useful in implementation as it will help the algorithm grow by learning about users. Building on the work of Kumar & Lease we also may want to consider using workers to validate their peers to ensure higher accuracy on batches were there is low accuracy.

Another step that Samasource can consider in order to improve accuracy while decreasing user cost is to include smart training into the accuracy algorithm. A smart training algorithm can use gold in order to predict patterns of worker error. If a worker dips below a quality threshold, the Samahub can use the predicted patterns in order to train the worker on problems they are consistently answering incorrectly within a dataset. In order to do this, however, Samasource will need to develop classifiers for their gold tasks, and match the errors that a user makers to the specific classifier in the gold. By recognizing errors within certain types of gold, the Samahub can then display a training round that teaches a worker how to do the task correctly. By including a smart classifier into the current Smart Gating Algorithm, Samasource will ensure that workers are learning from their mistakes, and eventually learn how to become better workers over time.

Works Cited:

Mason, W. A., & Watts, D. J. (2009). Financial incentives and the performance of crowds. In Proceedings of the ACM SIGKDD Workshop on Human Computation (pp. 77–85). New York: ACM.

Kumar, Abhimanu, and Matthew Lease. "Modeling annotator accuracies for supervised learning." Proceedings of the Workshop on Crowdsourcing for Search and Data Mining (CSDM) at the Fourth ACM International Conference on Web Search and Data Mining (WSDM). 2011.

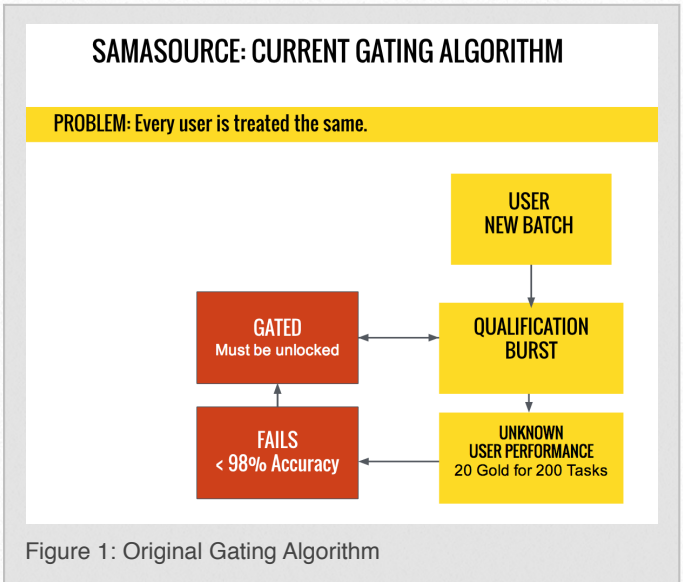

Figure 1: Original Gating Algorithm

Graphs:

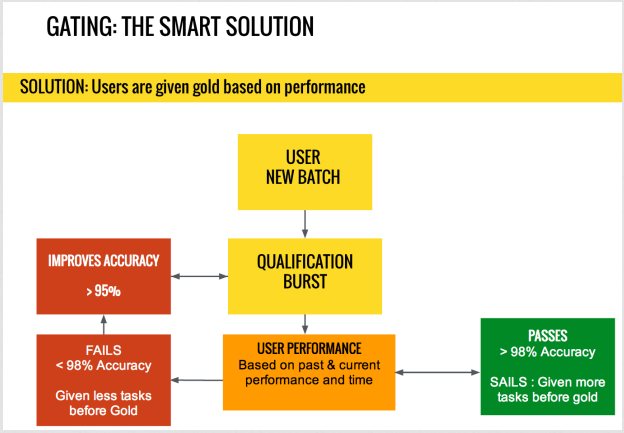

Figure 2: The Smart Gating Algorithm

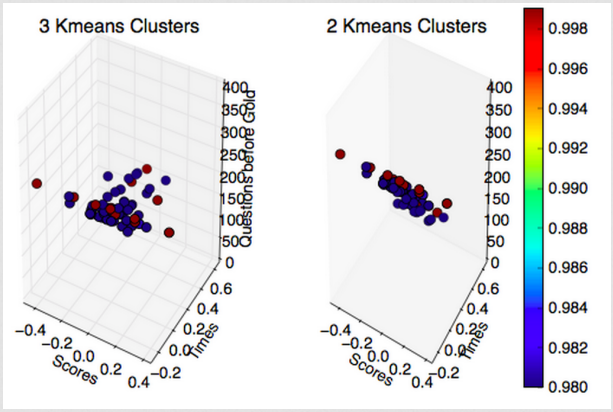

Figure 4

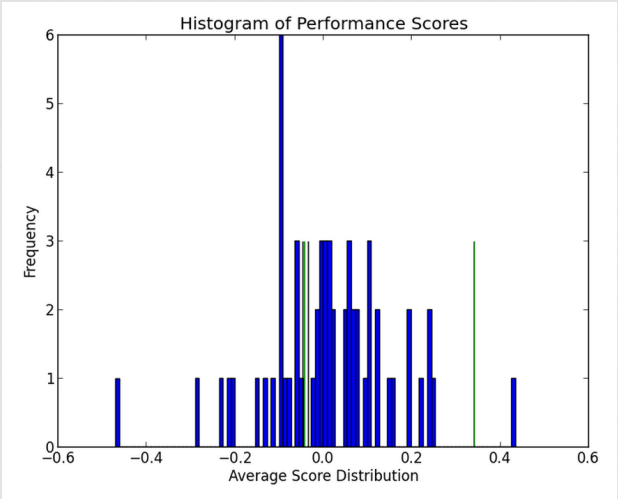

Figure 3

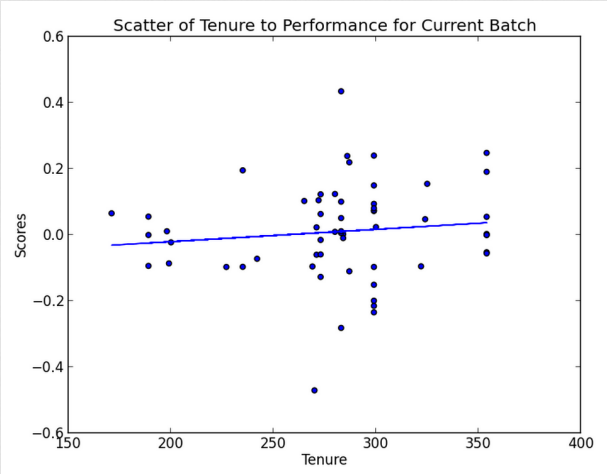

Figure 5