CrowdED: Guideline for Optimal Crowdsourcing Experimental Design

Amrapali Zaveri
Institute of Data Science, Maastricht University
Maastricht, Limburg, The Netherlands
amrapali.zaveri@maastrichtuniversity.nl

Pedro Hernandez Serrano
Institute of Data Science, Maastricht University
Maastricht, Limburg, The Netherlands
p.hernandezserrano@maastrichtuniversity.nl

Manisha Desai
Stanford University
Stanford, USA
manishad@stanford.edu

Michel Dumontier
Institute of Data Science, Maastricht University
Maastricht, Limburg, The Netherlands
michel.dumontier@maastrichtuniversity.nl

ABSTRACT

Crowdsourcing involves the creating of HITs (Human Intelligent Tasks), submitting them to a crowdsourcing platform and providing a monetary reward for each HIT. One of the advantages of using crowdsourcing is that the tasks can be highly parallelized, that is, the work is performed by a high number of workers in a decentralized setting. The design also offers a means to cross-check the accuracy of the answers by assigning each task to more than one person and thus relying on majority consensus as well as reward the workers according to their performance and productivity. Since each worker is paid per task, the costs can significantly increase, irrespective of the overall accuracy of the results. Thus, one important question when designing such crowdsourcing tasks that arise is how many workers to employ and how many tasks to assign to each worker when dealing with large amounts of tasks. That is, the main research questions we aim to answer is: 'Can we a-priori estimate optimal workers and tasks' assignment to obtain maximum accuracy on all tasks?'. Thus, we introduce a two-staged statistical guideline, CrowdED, for optimal crowdsourcing experimental design in order to a-priori estimate optimal workers and tasks' assignment to obtain maximum accuracy on all tasks. We describe the algorithm and present preliminary results and discussions. We implement the algorithm in Python and make it openly available on Github, provide a Jupyter Notebook and a R Shiny app for users to re-use, interact and apply in their own crowdsourcing experiments.

CCS CONCEPTS

• **Information systems → Crowdsourcing**; • **Human-centered computing → HCI design and evaluation methods**; **User models**; **HCI theory, concepts and models**; *User studies*; Human computer interaction (HCI); • **Applied computing →** *Life and medical sciences*;

KEYWORDS

crowdsourcing, biomedical, metadata, data quality, FAIR, reproducibility

ACM Reference Format:
Amrapali Zaveri, Pedro Hernandez Serrano, Manisha Desai, and Michel Dumontier. 2018. CrowdED: Guideline for Optimal Crowdsourcing Experimental Design. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3184558.3191543

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3191543

1 CROWDSOURCING AS A MEANS OF QUALITY ASSESSMENT

Enormous amounts of (biomedical) data have been and are being produced at an unprecedented rate by researchers all over the world. However, in order to enable this reuse, there is an urgent need to understand the structure of the experimental data, the conditions under which they were produced and the relevant information that other investigators may need to make sense of the data [4]. That is, there is a need for good quality i.e. structured, accurate and complete description of the data – defined as *metadata*. Good quality metadata is essential in finding, interpreting, and reusing existing data beyond what the original investigators envisioned. This, in turn, can facilitate a data-driven approach by combining and analyzing similar data to uncover novel insights or even more subtle trends in the data. These insights can then be formed into hypothesis that can be tested in the laboratory [11].

One of the means to assess the quality of this biomedical metadata that we propose is by the use of microtask crowdsourcing i.e. non-expert workers in order to reduce the cost and time involved for performing the same assessment by means of domain experts [8]. Crowdsourcing involves the creating of HITs (Human Intelligent Tasks), submitting them to a crowdsourcing platform (e.g. Amazon Mechanical Turk (MTurk)[1]) and providing a monetary reward for each HIT [8]. The tasks primarily rely on basic human abilities and natural language understanding but less on acquired skills such as domain knowledge. A great share of the tasks addressed via microtask platforms like MTurk could be referred to as 'routine tasks' - recognizing objects in images, transcribing audio and video material and text editing.

[1] http://mturk.com

One of the advantages of using crowdsourcing is that the tasks can be highly parallelized, that is, the work is performed by a high number of workers in a decentralized setting. The design also offers a means to cross-check the accuracy of the answers by assigning each task to more than one person and thus relying on majority consensus as well as reward the workers according to their performance and productivity. Since each worker is paid per task, the costs can significantly increase, irrespective of the overall accuracy of the results. Thus, one important question when designing such crowdsourcing tasks that arise is how many workers to employ and how many tasks to assign to each worker when dealing with large amounts of tasks. That is, how do we optimally design the task such that the right combination of workers and tasks can produce the maximum accuracy and can we determine this number a-priori.

In order to determine the number of workers as well as number of tasks that would be 'ideal' in order to solve the problem, we propose *CrowdED*, a two-staged *Crowd*sourcing *E*xperimental *D*esign. CrowdED provides a guideline for designing optimal crowdsourcing experiments. The main research questions we aim to answer is: *Can we a-priori estimate optimal workers and tasks' assignment to obtain maximum accuracy on all tasks?*

We describe the use case in section 2. We describe our two-staged statistical guideline, CrowdED in section 3. Preliminary results are reported in section 4. Related work is discussed in section 5. Finally, we conclude with an outlook on future work in section 6.

2 USE CASE: GEO METADATA

Amongst the several biomedical databases available on the Web, the Gene Expression Omnibus (GEO) is one of the largest, best-known biomedical databases [11]. GEO is an international public repository for high-throughput microarray and next-generation sequence functional genomic data submitted by the research community. The GEO database hosts >32, 000 public series (study records) submitted directly by 3, 000 laboratories, comprising 800, 000 samples derived from >1600 organisms (as of 2012). In GEO, a Sample Record describes the specific conditions under which an individual sample was handled, the manipulations it underwent, and the abundance measurement of each element derived from it.

In a sample, from the different metadata elements, we specifically chose the semi-structured 'characteristics' field, which contains information about, for example, the disease, strain, cell line etc. used in the study. This information is captured in a `key: value` pair format. Currently users can submit data to GEO via three ways: (i) spreadsheets, (ii) SOFT format (plain text) or (iii) MINiML format (XML). When users submit data to GEO via a spreadsheet (namely *GEOarchive spreadsheet*), it requires them to fill out a metadata template that follows the guidelines set out by the Minimum Information About a Microarray Experiment (MIAME) guidelines [1]. The metadata template includes fields for title, overall design, summary, the protocols (e.g. treatment, extraction, labeling, hybridization and data processing) as well as sample characteristics (e.g. organism, cell type, tissue). After submission, a curator checks the content and validity of the information provided [2]. This process is not only error-prone but also time consuming considering the amount of manual labor that is involved. Moreover, without a standardized set of terms with which to fill out the template fields, there are different versions of the same entity without any (semantic) links between them, thus leading to several quality issues.

Quality issues such as inaccuracy, inconsistency and incompleteness hamper the uptake of the datasets and also the reliability of the resultant applications making use of this data. All the 44,000,000+ `key: value` pairs in GEO suffer from these quality problems, which raises the scalability issue of performing large-scale curation. Additionally, with a scarcity of domain experts to curate the large amount of data in GEO, there is a need for more efficient methods for curating the metadata. Thus, we propose the use of crowdsourcing to perform metadata quality assessment.

We design a microtask as a classification task of identifying the correct category for a given key. These key categories belong to the top most frequently occurring keys in GEO. Additionally, five top most frequently occurring values are also provided to the worker. For example, the key e.g. 'disease specific survival years' along with five key categories, namely, 'cell line', 'disease', 'gender', 'strain' and 'time' is provided to the worker along with the values 8.22, 17.66, 4.51, 0.89 and 12.19. The worker's task is to choose 'one' of the categories that the given key (best) belongs to. In this example, the worker should choose 'time' as the correct answer since the values are numerical indicating a time period. However, with 44,000,000+ `key: value` pairs in GEO, we are faced the question of how many workers to employ in order to perform this large-scale curation and how many tasks to assign per worker so as to achieve maximum accuracy of consensus. This led to the design of CrowdED as described in the following section.

3 CROWDED

In this section, we describe the details of the two-staged design for which we provide guidance on choosing the optimal number of workers to obtain maximum accuracy for their experiment. Figure 1 provides an overview of the CrowdED guideline, which is divided into two stages. We assume a scenario where the worker's task (as described in section 2) is to choose *one* correct answer among *five* given key categories. There are two stages, where the first stage gathers information on tasks difficulty and worker competency. The second stage is then designed based on what is learned from the first stage.

3.1 Stage 1

In the first stage, the user (the requester) has the option to configure the following variables that represent the user's a priori assumptions. If unspecified, default (example) values are assumed, as specified in parentheses.

- No. of tasks (100)
- No. of workers (40)
- No. of tasks assigned to each worker (7)
- Proportion of easy (or hard) tasks (hard tasks - 0.2)
- Proportion of competent or so-called good (or less competent or poor) workers (competent workers - 0.8)
- Proportion of training tasks (0.4)

The number of workers per task is chosen such that it is an odd number, greater than the number of possible answers. This parameter is chosen in order to deal with cases with no consensus

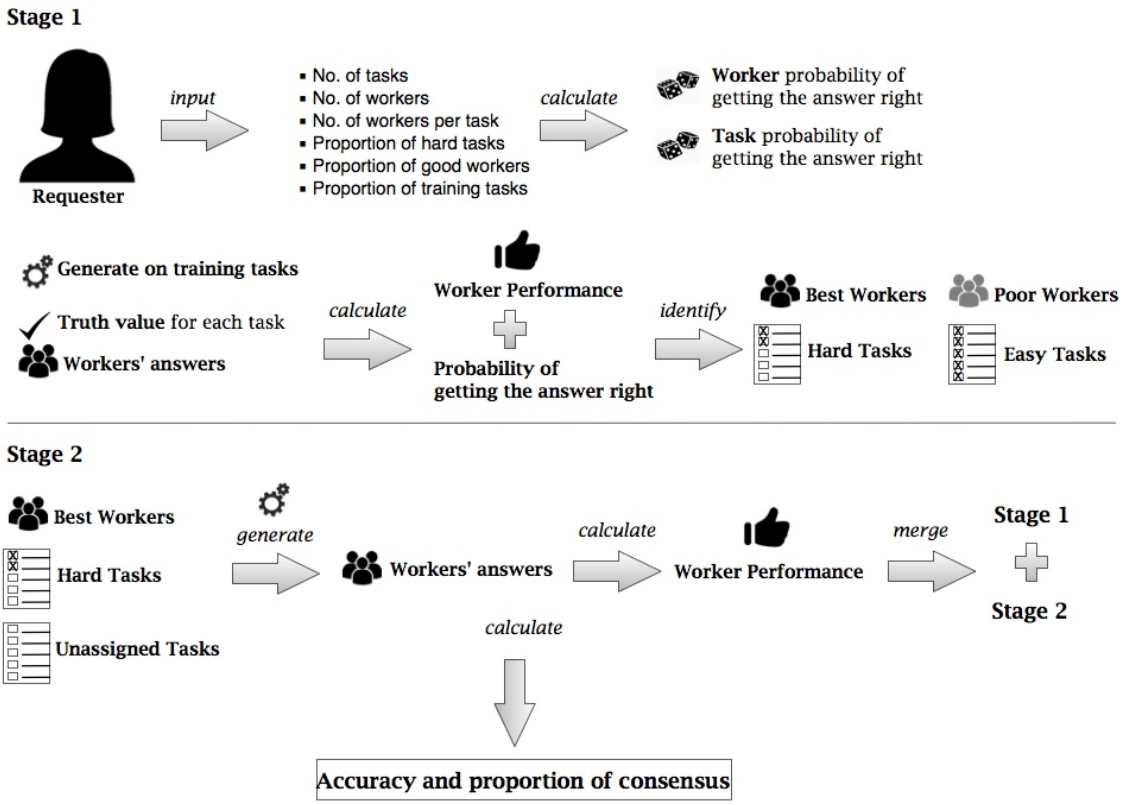

Figure 1: Overview of the CrowdED algorithm showing the steps involved in the two stages.

(e.g. if there are 5 possible answers and each of the 5 workers chooses a different answer). In our use case, the number of answers is 5, thus the number of workers per task is set to 7. However, each worker can have more than 7 tasks as we want to ensure that each of the (40% of the) training tasks have been evaluated by some of the workers. After setting the initial parameters, the algorithm randomly assigns (without replacement[2]) which tasks and workers are easy/hard and good/poor. Additionally, the true answer for each task is generated by randomly selecting from the set of answers such that they are evenly distributed across all the tasks. Next, the exact probabilities of each worker and each task of getting the answer right is calculated under the following assumptions.

$$p_w = \begin{cases} p_w \geq 3/4, & \text{if competent worker} \\ 1/2 < p_w < 3/4, & \text{if less competent worker} \\ 0, & \text{otherwise} \end{cases}$$

$$p_t = \begin{cases} p_t \geq 3/4, & \text{if easy task} \\ 1/2 < p_t < 3/4, & \text{if hard task} \\ 0, & \text{otherwise} \end{cases}$$

These values are chosen to represent variation in performance across workers. Based on the value specified for the proportion of tasks to train (40 in our example) the number of workers per task (7 in our example) are assigned to each task and the worker answer is generated.

In practice, true answers will not be known, and thus we rely on an agreement statistic to gauge the performance of the worker using the following metric: the average proportion of times a worker is in agreement with other workers for a given tasks over all tasks considered by the worker. The range of the performance value spans from 0 to 1. The values close to 1 indicate that the the worker had large consensus with other workers. Values close to 0 indicate that there was no consensus for that worker among other workers. Then, cut off values, above the median, of the performance of the worker and also the probability of getting the answer right is set to choose which workers get carried forward to Stage 2. The probability and the performance of the worker is combined since it is not always the case that the workers who had a high probability of getting the answer right in the beginning necessarily performed well in the actual tasks. Thus, this combination ensures that the *best* workers with high probabilities for both measures are identified. Additionally, for each task we determine whether it is an easy or hard task based on the workers' answers. That is, for all pairwise comparisons between the workers' answer the truth, we match how many pairs of workers arrived at the same answer for each task.

At the end of Stage 1, we get:

[2]https://www.ma.utexas.edu/users/parker/sampling/repl.htm

- Poor workers, those that did not achieve high consensus amongst other workers performing the same task.
  - These workers are flagged and not chosen for Stage 2.
- Best workers, those with a good performance value and assigned the good worker status at the beginning.
  - These workers are chosen for Stage 2.
- Easy tasks, those that have the predicted performance to be 3/4 to 1.
  - These tasks are considered to have achieved majority consensus and are not carried forward to Stage 2 for re-assessment.
- Hard tasks, those that have the predicted performance to be below 3/4. That is, those that did not achieve majority consensus in Stage 1.
  - These tasks are then carried forward to Stage 2 to be re-assessed.
  - Unassigned tasks
    - The total number of tasks (100) minus the proportion of tasks to train (40) = 60.

3.2 Stage 2

In this stage, the algorithm assigns the hard and unassigned tasks to the best workers, generates the workers' answers and calculates the overall accuracy of all the tasks. Stage 2 begins with:

- Best workers
- Hard tasks
- Unassigned tasks

Before the best workers are assigned to the remaining of the tasks, it should be ensured that the workers do not perform the same tasks that they were assigned in Stage 1. To ensure this, the following pseudo-code is used:

```
for each task
    select odd number of workers
        check if the workers already done this task
        and exclude them
        calculate how many tasks left
      while the number of workers is the same as
      the number of workers per task variable
       re-select this number of workers
```

Then, the worker answers are generated, as described in Stage 1. Next, the performance of each of the workers is calculated. Finally, data from all the tasks from Stage 1 and Stage 2 are merged to get the final dataset of all tasks and all workers. After merging the datasets, a final answer is assigned to each task based on the majority consensus of the workers' answers[3]. Additionally, the proportion of tasks for which the workers got the right answer is also calculated.

Finally, the accuracy of all the tasks as well as the workers is calculated using the formula described below.
Let T = total number of tasks in the experiment
Let $\widehat{t_i}$ = (number of workers correctly answered task t_i) / (number of workers doing task t_i) considering $\widehat{t_i} \in [0, 1]$
Let C = subset of tasks which achieved consensus is greater than $\frac{1}{2}$, more than a half means a majority.
Let n_C = number of elements in the subset$C$.
Then the subset C is defined as follows:

$$C = \{\forall \ \widehat{t_i} \ \mid \ \widehat{t_i} > \frac{1}{2}\}$$

Finally, the accuracy of consensus is a combination of the following two statistics of the subset C.
Mean of consensus:

$$\widehat{a} = \frac{1}{n_C} \sum_{t_i \in C} \widehat{t_i}$$

Proportion of consensus:

$$\widehat{p} = \frac{n_C}{T}$$

The proportion of consensus can be seen as the percentage of tasks under consensus and the mean of consensus is how accurate the consensus is. These consensus values help determine the accuracy and thus the optimal number of workers one needs for the total number of tasks.

3.3 Implementation

The algorithm is written in Python and openly available for reuse at *https://github.com/pedrohserrano/crowdED*. A Python package is available at *https://pypi.python.org/pypi/crowdED* (requires Python 3 or later versions) where one can use CrowdED to test with one's own values. A Jupyter Notebook version is available[4] where one can see the exact steps of CrowdED. Additionally, we provide a user interface at *https://pedrohserrano.shinyapps.io/crowdapp/*, built using the R Shiny apps[5] (depicted in Figure 5) to visualize the interaction of the variables and their effects on the overall accuracy. Even though at this stage, the app does not allow direct user input (which is part of the future work), in the 'Analysis' tab, one can vary the number of simulations to see the effect on accuracy in the form of graphs.

4 PRELIMINARY RESULTS

We tested our algorithm by generating random value distributions for the variables as:

- tasks = [60, 80, 100, 120, 140, 160, 180]
- workers = [20, 30, 40]
- answers key = ["liver", "blood", "lung", "brain", "heart"]
- good workers = [0.1, 0.3, 0.5, 0.7, 0.9]
- hard tasks = [0.1, 0.3, 0.5, 0.7, 0.9]
- proportion of training tasks = [0.2, 0.3, 0.4, 0.5, 0.6]
- workers per task = [3, 5, 7, 9, 11]

In total, there were 13, 125 of combinations that were tested for each of the variables, and every combination were simulated one thousand times, preliminary results of which are described below.

[3] This is done because there may be cases where the workers converge on an answer different than the truth value, which need not be necessarily incorrect.

[4] https://github.com/pedrohserrano/crowdED/blob/master/notebooks/Crowdsourcing.ipynb

[5] https://shiny.rstudio.com/

Number of Tasks	Number of Workers	Proportion of Good Workers				
		10 %	30 %	50 %	70 %	90 %
60	20	0.87661	0.85961	0.85384	0.86145	0.85315
	30	0.86996	0.85055	0.85144	0.85299	0.84045
	40	0.85791	0.83815	0.83287	0.83379	0.83489
80	20	0.86493	0.85461	0.86413	0.85244	0.85301
	30	0.86095	0.85037	0.84459	0.85513	0.83438
	40	0.85810	0.84088	0.83334	0.82565	0.82112
100	20	0.88080	0.86022	0.86686	0.85735	0.83548
	30	0.86401	0.85622	0.84492	0.83450	0.82601
	40	0.84515	0.84897	0.83519	0.84463	0.82794
120	20	0.88020	0.86836	0.83587	0.83088	0.86988
	30	0.87205	0.85328	0.84353	0.83614	0.84371
	40	0.85294	0.84713	0.83863	0.82909	0.83094
140	20	0.87812	0.87120	0.85982	0.84335	0.85139
	30	0.86736	0.85167	0.84562	0.84252	0.84435
	40	0.85914	0.84436	0.83974	0.83157	0.82643
160	20	0.86765	0.86634	0.85470	0.85452	0.85003
	30	0.86482	0.85353	0.84452	0.84129	0.83758
	40	0.85407	0.84838	0.83760	0.82763	0.83091
180	20	0.86435	0.85630	0.85946	0.85503	0.84408
	30	0.86246	0.84912	0.84387	0.84286	0.83549
	40	0.85462	0.84237	0.83476	0.82911	0.83056

(a) Proportion of good workers

Number of Tasks	Number of Workers	Proportion of Hard Tasks				
		10 %	30 %	50 %	70 %	90 %
60	20	0.8962	0.9008	0.8488	0.8434	0.8107
	30	0.9032	0.8747	0.8497	0.8247	0.8074
	40	0.8895	0.8674	0.8452	0.8104	0.7791
80	20	0.8991	0.8772	0.8484	0.8392	0.8155
	30	0.8974	0.8743	0.8546	0.8181	0.7955
	40	0.8916	0.8657	0.8399	0.8089	0.7839
100	20	0.9018	0.8830	0.8597	0.8261	0.8156
	30	0.8952	0.8709	0.8487	0.8236	0.7978
	40	0.8914	0.8664	0.8419	0.8164	0.7869
120	20	0.8992	0.8880	0.8600	0.8264	0.7992
	30	0.8974	0.8686	0.8559	0.8241	0.7966
	40	0.8904	0.8625	0.8444	0.8128	0.7833
140	20	0.9017	0.8781	0.8594	0.8463	0.8082
	30	0.8956	0.8759	0.8503	0.8261	0.7961
	40	0.8931	0.8687	0.8451	0.8102	0.7807
160	20	0.9043	0.8813	0.8562	0.8447	0.8051
	30	0.8971	0.8715	0.8488	0.8279	0.7940
	40	0.8935	0.8709	0.8395	0.8149	0.7829
180	20	0.9064	0.8755	0.8551	0.8387	0.8080
	30	0.8938	0.8758	0.8491	0.8205	0.7931
	40	0.8897	0.8673	0.8414	0.8086	0.7866

(b) Proportion of hard tasks

Figure 2: (a) Matrix showing accuracy values for different number of tasks, number of workers and varying proportions of the good workers. The darker green cells show higher accuracy while the blue ones show lower accuracy. Results suggest that starting out with good workers does not always lead to high accuracy. The performance of the workers in combination with whether they were a good worker ensures that they are the best workers. This is why we need a two-staged crowdsourcing design. (b) Matrix showing accuracy values for different number of tasks, number of workers and varying proportions of the hard tasks. The darker green cells show higher accuracy while the blue ones show lower accuracy. Results support our intuition that the lesser the hard tasks (10%), the higher the accuracy.

Proportion of good and poor workers. In most crowdsourcing platforms, one has the option to choose 'good' workers before launching the tasks. For example, in Mturk the so-called 'Master workers' can be chosen by specifying their HIT (Human Intelligence Task) acceptance rate. These workers are assigned this status depending on their performance over all the tasks they have attempted and their acceptance rate for these. However, based on our results, we observe that starting out with good workers does not always lead to high accuracy. Figure 2(a) shows a matrix with the accuracy values for different number of tasks, number of workers and varying proportions of the good workers. The darker green cells show higher accuracy while the blue ones show lower accuracy. With 90% of good workers at the start, the accuracy ranges from 0.82 to 0.86 whereas starting with 10% of the tasks, the accuracy ranges from 0.84 to 0.88. Thus, it is inconclusive of the proportion of good workers to start with. However, adopting the two-staged algorithm ensures that only the best workers are chosen to perform all the tasks. Therefore, calculating the performance of the workers in combination with whether she was a good worker (from the beginning) ensures that she is the best worker. This is why we need a two-staged crowdsourcing design in order to test the workers performance and choosing only the best workers to perform the total set of tasks in order to achieve high accuracy.

Proportion of easy and hard tasks. We determined the effect on accuracy depending on the proportion of hard and easy tasks. Figure 2(b) shows a matrix of the accuracy values for different number of tasks, number of workers and varying proportions of the hard tasks. The darker green cells show higher accuracy while the blue ones show lower accuracy. With 10% of hard tasks, the accuracy ranges from 0.88 to 0.9 whereas with 90% of hard tasks, the accuracy ranges from 0.78 to 0.8. Results support the intuition that reduced difficulty (10%) in tasks result in higher accuracy.

Proportion of training tasks. We analyzed the results for the ideal proportion of total tasks that should be trained in Stage 1. Figure 3 shows a heat map with the accuracy values for different workers per task and the percentage of tasks trained in Stage 1. The darker green bubbles show higher accuracy while the blue ones show lower accuracy. The values inside the bubble are 'a' is mean and 'p' is proportion of consensus (as described in Section 3). With 20%, 30%, 40% of training tasks and 3, 5, 7 and 9 workers per task, the accuracy 'a' is lower as compared to 40%, 50%, 60% of training tasks with 3, 5 and 7 workers per task. Results suggest that ideal is to use

3, 5 or 7 workers per task and train 40% to 60% of the task in Stage 1 to achieve high accuracy.

Number of workers per task. We examined how the number of workers per task affects the accuracy and the proportion of consensus. Figure 4 shows how the ratio of all workers over all tasks (X-axis) compares to proportion of accuracy and proportion of consensus (Y-axis) when the number of workers is different per task (3, 5, 7 and 9). With 3, 5 and 7 workers per task, the accuracy of consensus remains stable at a range of 0.8 to 0.9. However, the accuracy declines significantly with 9 workers per task. Additionally, the proportion of consensus increases uniformly along with significant p-values for each variation with 3, 5 and 7 workers per task. However, with 9 workers per task, the proportion for accuracy also decreases along with a non-significant p-value. Results suggest that after 9 workers per task the accuracy and proportion of consensus decreases.

Overall results. The preliminary results of these simulations suggest that in order to achieve high accuracy:

- the number of workers should be 40% to 60% of the total number of tasks
- to train workers on 40% to 60% of the tasks in Stage 1
- to set the number of workers per task to be either 3, 5 or 7 (or fewer than 9)
- to reduce the number of hard tasks
- to adopt the two-staged algorithm to identify the best workers

5 RELATED WORK

There have been empirical studies to determine the 'optimal' number of workers per task. However, these studies only focus on their domain or task at hand. For example, there is an adaptive model [10] which studied different scenarios of increasing complexity of tasks wrt. the worker quality. This strategy was applied particularly to labeling tasks. However, they assume that all workers are of the same quality Another strategy employs active learning algorithms (changing the assignments per tasks in real-time) to minimize the number of questions asked to the crowd to maximize the number of tasks [9]. However, reportedly this model is extremely expensive to adapt in a real-world experiment. Another study assigns tasks based on the quality of workers and suggest that, for example, between three and eight workers is ideal [3]. In [13], test questions created from a generalized knowledge base are used to estimate the reliability of the new workers. Their result suggest that this approach performs better than using gold-standard tasks automated selection of knowledge base questions for quality control [12] used a hybrid approach of self-rating and gold-standard task for estimating the expertise of workers, however the self-assessment does not ensure high accuracy on the actual tasks.

Two models [5] and [6] provided approaches for cost-quality and cost-time optimization respectively. However, the former model is focused on each task and requires that the pay be set based on progress of the total number of tasks. The latter model assumes a fixed number of workers per task and and does not optimize quality by taking a variable number of workers based on each task difficulty. A recent study [7] introduced an AI agent, OCTOPUS, to jointly balance the quality of work, total cost incurred and time to completion and significantly outperformed existing state-of-the-art approaches. However, OCTOPUS only tested for tasks that contained a binary choice for the answer. CrowdED is distinct from all these studies as it offers a two-staged statistical model that can *a-priori* estimate the number of workers to assign per task in order to gain maximum accuracy whereas OCTOPUS optimizes on-the-fly.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we describe a two-staged statistical guideline, CrowdED, for designing optimal crowdsourcing tasks in order to a-priori estimate optimal workers and tasks' assignment to obtain maximum accuracy on all tasks. We implemented the algorithm in Python and made it openly available on Github, a Python package, provided a Jupyter Notebook and a R Shiny app for users to re-use, interact and apply in their own crowdsourcing experiments. Our preliminary results suggest the 'optimal' values for each of the variables in order to achieve maximum accuracy for the tasks. This is a first step towards answering our research question of estimate a-priori the optimal task-worker assignment towards high accuracy.

At this stage, CrowdED only simulates multiple-choice questions type of crowdsourcing experiments and not free text answers. In future work, we will explore the feasibility of Natural Language Processing (NLP) approaches to evaluate the accuracy of free-text answers. Also as part of future work, we will assess the operating characteristics of this design, and perform testing of the algorithm on our use case as well as other real-world input data. Additionally, we will compare the results of these approaches to the baseline approaches that are standard crowdsourcing platforms (e.g. CrowdFlower[6], MTurk). Moreover, we will account for the budgetary constraints in the optimization algorithm. Also, we will extend the interface such that a user can vary parameters and assumptions to see how sensitive the design is to various assumptions.

7 ACKNOWLEDGEMENTS

The authors would like to acknowledge that this project was funded by NCATS (National Center for Advancing Translational Science https://ncats.nih.gov/) Deeplink grant (no. 35700002N).

REFERENCES

[1] Brazma A, Hingamp P, Quackenbush J, Sherlock G, Spellman P, Stoeckert C, Aach J, Ansorge W, Ball CA, Causton HC, Gaasterland T, Glenisson P, Holstege FC, Kim IF, Markowitz V, Matese JC, Parkinson H, Robinson A, Sarkans U, Schulze-Kremer S, Stewart J, Taylor R, Vilo J, and Vingron M. 2011. Minimum information about a microarray experiment (MIAME)-toward standards for microarray data. *Nature Genetics* 29 (2011), 365 – 371. Issue 4.

[2] Tanya Barrett, Dennis B. Troup, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Rolf N. Muertter, Michelle Holko, Oluwabukunmi Ayanbule, Andrey Yefanov, and Alexandra Soboleva. 2011. NCBI GEO: archive for functional genomics data sets âĂŤ 10 years on. *Nucleic Acids Research* 39 (2011), 991 – 995.

[3] Good BM, Nanis M, Wu C, and Su AI. 2015. Microtask crowdsourcing for disease mention annotation in PubMed abstracts. *Pac Symp Biocomput.* (2015), 282–293.

[4] C. L. Borgman. 2012. The conundrum of sharing research data. *Journal of the American Society for Information Science and Technology* 63 (2012), 1059 âĂŞ 1078. Issue 6.

[5] Peng Dai, Christopher H. Lin, Mausam, and Daniel S. Weld. 2013. POMDP-based Control of Workflows for Crowdsourcing. *Artif. Intell.* 202, 1 (Sept. 2013), 52–85. https://doi.org/10.1016/j.artint.2013.06.002

[6] crowdflower.com

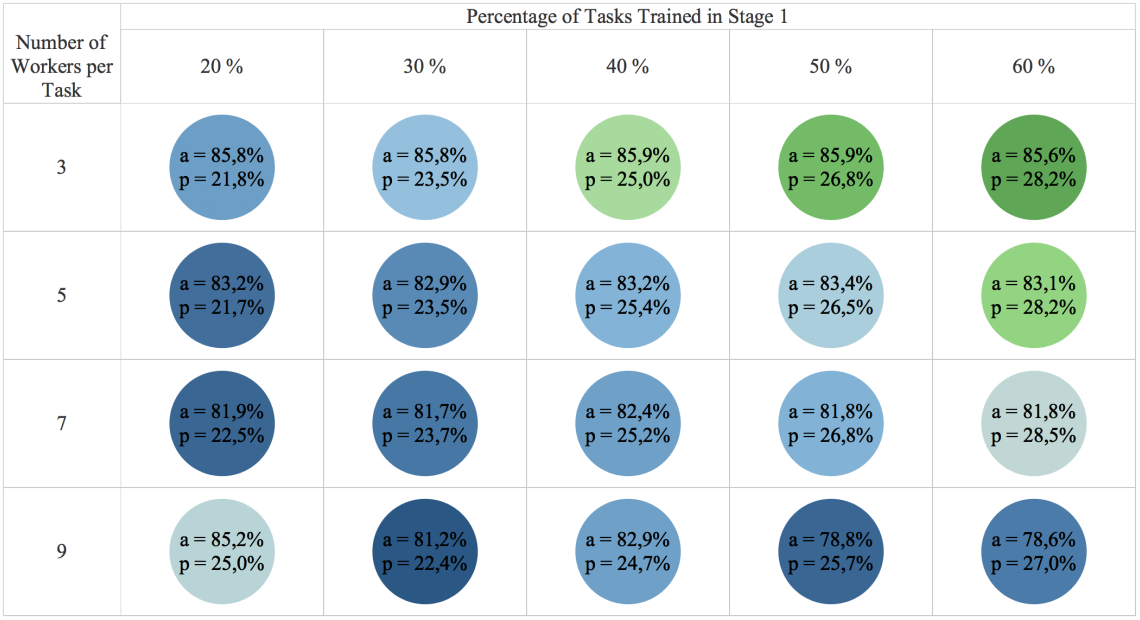

Figure 3: Heat map showing the accuracy for different workers per task vs. the percentage of tasks trained in Stage 1. Green bubbles show higher accuracy while the blue bubbles indicate lower accuracy. The values inside the bubble are 'a' is accuracy of consensuses and 'p' is proportion of consensus. Results suggest that ideal is to use 3, 5 or 7 workers per task and train 40% to 60% of the task in Stage 1 to achieve high accuracy.

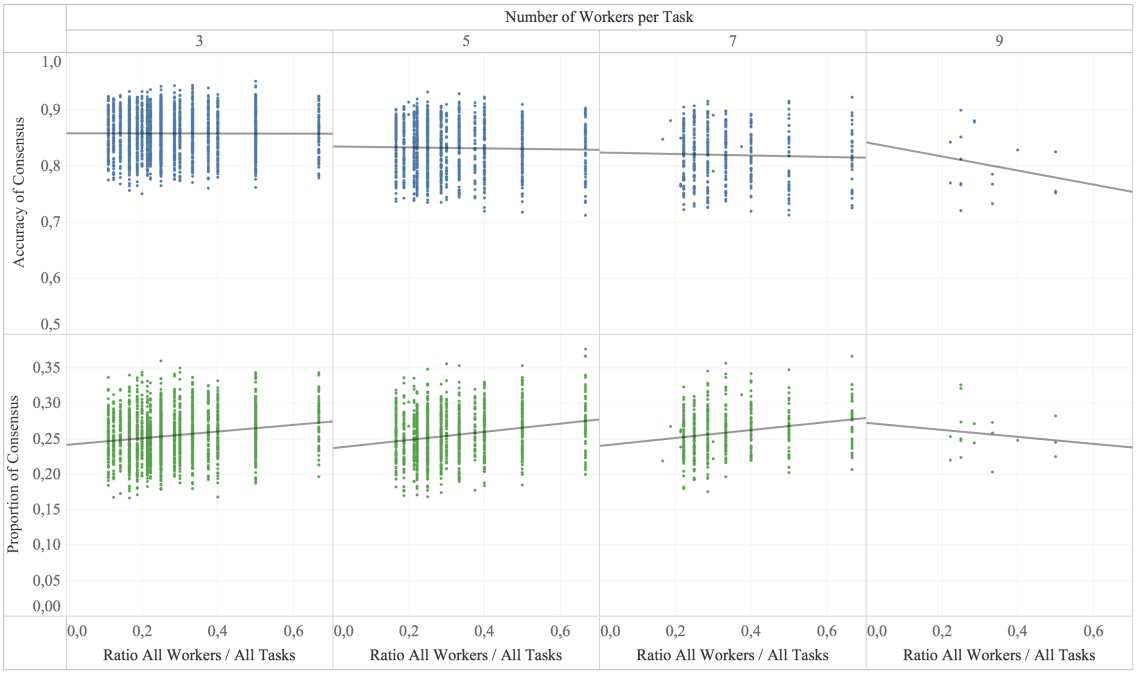

Figure 4: Plots of the ratio of all workers over all tasks (X-axis) with respect to the proportion of accuracy (above) and proportion of consensus (below) on the Y-axis for varying number of workers per task (3, 5, 7 and 9).

[6] Yihan Gao and Aditya Parameswaran. 2014. Finish Them!: Pricing Algorithms for Human Computation. *Proc. VLDB Endow.* 7, 14 (Oct. 2014), 1965–1976. https://doi.org/10.14778/2733085.2733101

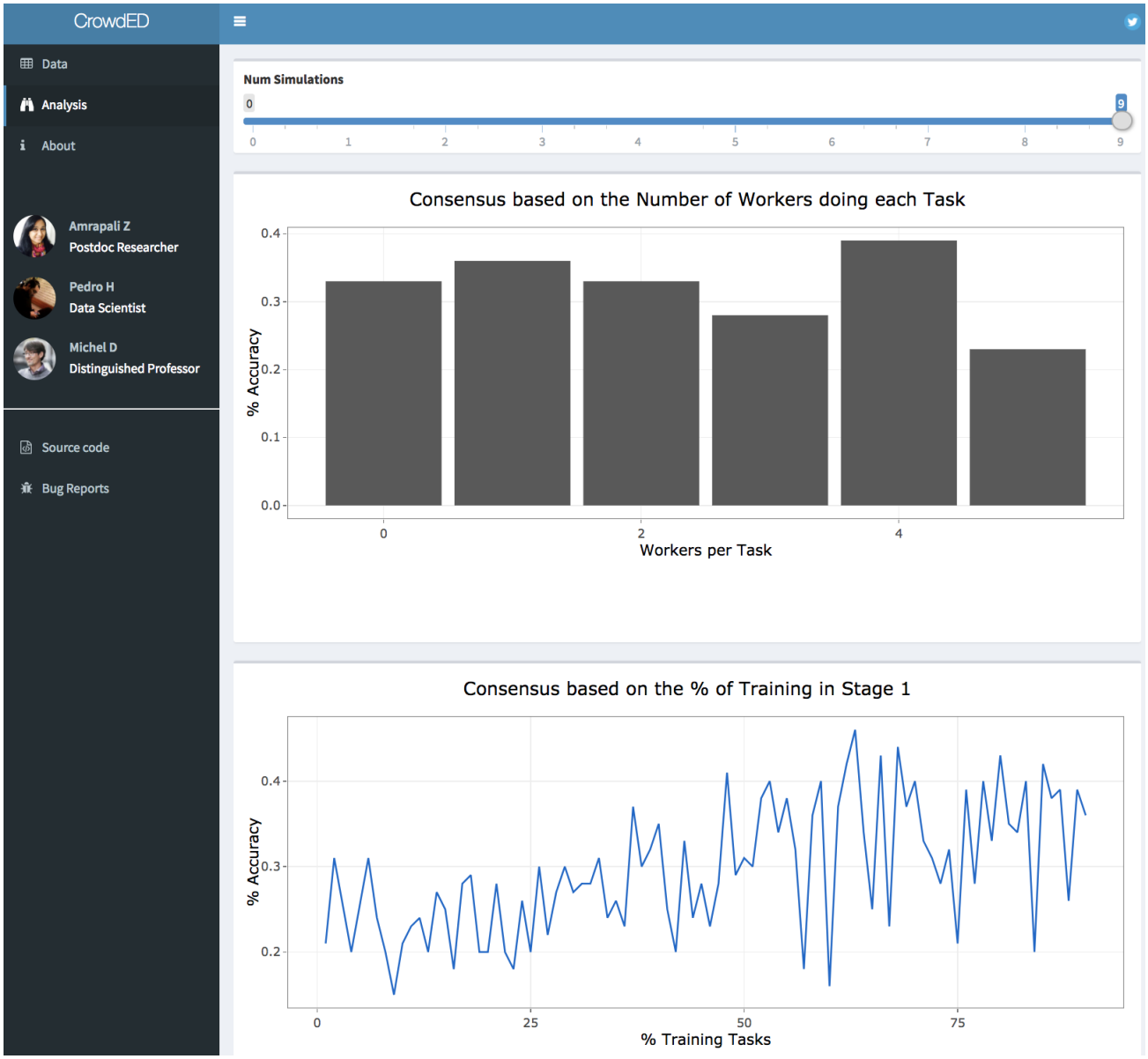

Figure 5: Screenshot of the CrowdED R Shiny app showing a beta version of the interface available at https://pedrohserrano.shinyapps.io/crowdapp/. The 'Analysis' tab is pre-configured to the default values and provides a slider for 10 simulations. The accuracy calculated for each simulation is depicted as graphs.

[7] Karan Goel, Shreya Rajpal, and Mausam. 2017. Octopus: A Framework for Cost-Quality-Time Optimization in Crowdsourcing. *CoRR* abs/1702.03488 (2017). arXiv:1702.03488 http://arxiv.org/abs/1702.03488

[8] Jeff Howe. 2006. The Rise of Crowdsourcing. *Wired Magazine* 14, 6 (06 2006). http://www.wired.com/wired/archive/14.06/crowds.html

[9] Barzan Mozafari, Purna Sarkar, Michael Franklin, Michael Jordan, and Samuel Madden. 2014. Scaling up crowd-sourcing to very large datasets: a case for active learning. *Proceedings of the VLDB Endowment* 8 (2014), 125–136. Issue 2.

[10] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. 2008. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 614 – 622.

[11] Barrett T, Wilhite SE, Ledoux P, Evangelista C, Kim IF, Tomashevsky M, Marshall KA, Phillippy KH, Sherman PM, Holko M, Yefanov A, Lee H, Zhang N, Robertson CL, Serova N, Davis S, and Soboleva A. 2013. NCBI GEO: archive for functional genomics data sets–update. *Nucleic Acids Research* 41 (2013), 991 – 995.

[12] U. Ul Hassan, S. O'Riain, and E. Curry. 2013. Effects of expertise assessment on the quality of task routing in human computation. *Proceedings of the 2nd International Workshop on Social Media for Crowdsourcing and Human Computation* (2013).

[13] Umair ul Hassan, Amrapali Zaveri, Edgard Marx, Edward Curry, and Jens Lehmann. 2016. ACRyLIQ: Leveraging DBpedia for Adaptive Crowdsourcing in Linked Data Quality Assessment. In *Knowledge Engineering and Knowledge Management*, Eva Blomqvist, Paolo Ciancarini, Francesco Poggi, and Fabio Vitali (Eds.). Springer International Publishing, Cham, 681–696.