Crowdsourcing from Scratch:
A Pragmatic Experiment in Data Collection by Novice Requesters

Alexandra Papoutsaki, Hua Guo, Danae Metaxa-Kakavouli,
Connor Gramazio, Jeff Rasley, Wenting Xie, Guan Wang, and **Jeff Huang**
Department of Computer Science
Brown University

Abstract

As crowdsourcing has gained prominence in recent years, an increasing number of people turn to popular crowdsourcing platforms for their many uses. Experienced members of the crowdsourcing community have developed numerous systems both separately and in conjunction with these platforms, along with other tools and design techniques, to gain more specialized functionality and overcome various shortcomings. It is unclear, however, how novice requesters using crowdsourcing platforms for general tasks experience existing platforms and how, if at all, their approaches deviate from the best practices established by the crowdsourcing research community. We conduct an experiment with a class of 19 students to study how novice requesters design crowdsourcing tasks. Each student tried their hand at crowdsourcing a real data collection task with a fixed budget and realistic time constraint. Students used Amazon Mechanical Turk to gather information about the academic careers of over 2,000 professors from 50 top Computer Science departments in the U.S. In addition to curating this dataset, we classify the strategies which emerged, discuss design choices students made on task dimensions, and compare these novice strategies to best practices identified in crowdsourcing literature. Finally, we summarize design pitfalls and effective strategies observed to provide guidelines for novice requesters.

Introduction

Crowdsourcing has gained increasing popularity in recent years due to its power in extracting information from unstructured sources, tasks that machine learning algorithms are unable to perform effectively. More people with little prior knowledge of crowdsourcing are getting involved, thus it is important to understand how novice requesters design crowdsourcing tasks to better support them. In this paper, we describe an experiment in which we observed and analyzed how novice requesters performed a non-contrived crowdsourcing task using crowdsourcing. The task was to collect information on the academic development of all faculty in 50 highly-ranked Computer Science departments—over 2,000 faculty in total. Through this experiment, we identify and compare six distinct crowdsourcing strategies.

We believe this is the first empirical analysis of how novice crowdsourcing requesters design tasks. This experiment was conducted as a class assignment based on a realistic crowdsourcing scenario: students were provided a fixed sum of money to compensate workers, and a limited amount of time to complete the assignment. They came up with different crowdsourcing strategies, which varied across multiple dimensions, including size of task, restriction of input, and amount of interaction with workers. The curated dataset is available as a public resource[1] and has already been visited by thousands of researchers.

This paper blends a quantitative analysis with qualitative interpretations of the students' observations. We believe our approach was fruitful in comparing different dimensions and providing interpretation. These findings have valuable implications for novice requesters when designing crowdsourcing jobs, especially for data collection. One finding was that novice requesters do not successfully integrate verification strategies on their jobs or tend to ignore this stage until they are well beyond their budget.

Our contributions are twofold. First, we report strategies, as well design choices that emerged when novice requesters designed crowdsourcing tasks for a realistic data collection problem. We discuss the observed rationales and difficulties behind task designs, relating them to systems and best practices documented in the research community, and summarize implications for how to better support novice requesters. Second, we compare how task results vary across these design choices to derive task design guidelines for novice requesters. These guidelines will give readers a better sense of how they might structure jobs to maximize results.

Related Work

We discuss existing crowdsourcing taxonomies that inspired our classification of strategies and dimensions; then relate to previous work on developing tools, techniques, and systems for supporting crowdsourcing requesters.

Taxonomies and Categorizations of Crowdsourcing

Researchers from a diversity of fields have generated taxonomies for a variety of crowdsourcing platforms, systems, and tasks. Geiger et al. created a four-dimensional taxonomy for crowdsourcing strategies based on 46 crowdsourcing examples (Geiger et al. 2011). Their dimensions were:

[1] http://hci.cs.brown.edu/csprofessors/

(1) preselection of contributors; (2) aggregation of contributions; (3) remuneration for contributions; and (4) accessibility of peer contributions. Other work has similarly deconstructed crowdsourcing strategies into separate dimensions (Malone, Laubacher, and Dellarocas 2010). We consider many of these these attributes of crowdsourcing in our evaluation of strategies used by novice requesters.

Others have established classifications of different crowdsourcing tasks (Corney et al. 2009; Quinn and Bederson 2011). To remove this extra dimension of complexity, we restrict the specific type of task to a particular data collection task in order to focus on novice requester strategies.

Researchers have also studied relevant components of crowdsourcing systems, including the degree of collaboration between workers, worker recruitment, and manual involvement by researchers (Hetmank 2013). Specifically, Doan et al. describe nine dimensions including nature of collaboration and degree of manual effort (Doan, Ramakrishnan, and Halevy 2011). We purposefully focus on the existing Mechanical Turk (MTurk) interface rather than an external or novel system, particularly as it is used by novice requesters, and evaluate it with regard to each dimension.

Existing Tools, Techniques, and Systems

Existing research has led to the development of a plethora of specialized crowdsourcing systems both for use by novices and those with more expertise. In particular, many systems have been developed to make crowdsourcing more effective for specific use cases and experienced requesters. Systems such as CrowdDB and Qurk use the crowd for a specific purpose: performing queries, sorts, joins, and other database functions (Franklin et al. 2011; Marcus et al. 2011). Similarly, Legion:Scribe is a specialized system for a particular task: helping non-expert volunteers provide high-quality captions (Lasecki et al. 2013). CrowdForge is an example of a more general-purpose system, which allows requesters to follow a partition-map-reduce pattern for breaking down larger needs into microtasks (Kittur et al. 2011). Precisely because many specialized systems for informed requesters have been developed, our work seeks to focus on novice requesters doing a generalizable data collection task.

Systems have also been developed explicitly for the purpose of bringing the benefits of crowd-powered work to an inexperienced audience. One of the best-known examples, the Soylent system, integrates crowd-work into a traditional word processor to allow users to benefit from crowdsourcing without needing a substantial or sophisticated understanding of the technique (Bernstein et al. 2010). In particular, the Find-Fix-Verify strategy built into Soylent provides users with high-quality edits to their documents automatically. Another system, Fantasktic, is built to improve novices' crowdsourcing results by providing a better submission interface, allowing requesters to preview their tasks from a worker's perspective, and automatically generating tutorials to provide guidance to crowdworkers (Gutheim and Hartmann 2012). Each of these novice-facing systems is built to address a common mistake novices make when crowdsourcing, such as inexperience verifying data or failing to provide sufficient guidance to workers. In contrast, our work seeks to holistically study novices' processes, successes, and common mistakes when using crowdsourcing. These insights are both valuable for further development of robust systems for novice requesters, but also for novice requesters who may not be aware of existing systems or may want to use crowdsourcing for a more general data collection task, one for which existing systems are irrelevant.

To aid in the creation of complex crowdsourcing processes, researchers have also developed various assistive tools. TurKit introduced an API for iterative crowdsourcing tasks such as sorting (Little et al. 2010). This tool is of particular use to experienced programmers looking to programmatically integrate crowdsourcing into computational systems. Another task-development tool, Turkomatic, uses crowdsourcing to help the design of task workflow (Kulkarni, Can, and Hartmann 2012). We identified similar creative uses of the crowd in meta-strategies such as this one when observing the crowdsourcing behaviors of novice requesters.

Existing research has also studied useful techniques and design guidelines for improving crowdsourcing results. Faridani et al. recommend balancing price against desired completion time, and provide a predictive model illustrating this relationship (Faradani, Hartmann, and Ipeirotis 2011). Other work has compared different platforms, showing the degrees to which intrinsic and extrinsic motivation influence crowdworkers (Kaufmann, Schulze, and Veit 2011). Researchers have also developed recommendations for improving exploratory tasks (Willett, Heer, and Agrawala 2012; Kittur, Chi, and Suh 2008) and have built models to predict quality of results based the task's relevant parameters, including the size of each task, number of workers per task, and payment (Huang et al. 2010). Since novice requesters are unlikely to consistently and effectively implement best practices when designing their applications of crowdsourcing, this work studies novice practices in order to provide recommendations for novice users and for crowdsourcing platforms like MTurk seeking to become more novice-friendly.

The Experiment

Nineteen undergraduate and graduate students completed an assignment as part of a Computer Science Human-Computer Interaction seminar. Students had no previous experience with crowdsourcing, but had read 2 seminal papers that introduced them to the field (Kittur, Chi, and Suh 2008; Bernstein et al. 2010). Each student was given $30 in Amazon credit and was asked to come up with one or more data collection strategies to use on MTurk. The goal as a class was to create a full record of all Computer Science faculty in 50 top Computer Science graduate programs, according to the popular US News ranking[2]. We chose this experiment because it involves a non-trivial data collection task with some practical utility, and contains realistic challenges of subjectivity, varying difficulty of finding the data, and even data that are unavailable anywhere online. Each student was responsible for compiling a complete record on faculty at

[2] http://www.usnews.com/best-graduate-schools

5–10 universities to ensure a fair division between departments with different sizes, resulting in two complete records per university. Ten pieces of information were collected for each faculty member: their full name, affiliated institution, ranking (one of Full, Associate, Assistant), main area of research (one of twenty fields, according to Microsoft Academic Research[3]), the year they joined the university, where they acquired their Bachelors, Masters, and Doctorate degrees, where they did their postdoc, and finally, links to pages containing the aforementioned information.

Collecting information about faculty is a challenging task since the data must be gathered from a variety of sources in different formats. Additionally, a certain degree of knowledge on Computer Science or even general academic terminology is required, posing a challenge to workers with no academic background or familiarity with relevant jargon. This task becomes even harder for the requester since we imposed realistic financial and time constraints on the crowdsourcing work by setting a fixed budget and deadline. Due to its challenging nature, we believe our task is representative of many free-form data collection and entry tasks, while simultaneously providing an interesting and useful dataset. Finally, students can be seen as typical novices as they lack experience in crowdsourcing, but are motivated to collect accurate data as a portion of their final grade depended on it.

After 16 days the students amassed about 2,200 entries of faculty, along with individual project reports that spanned between 10 to 20 pages, containing the data collection strategies used along with reflections on successes and failures. All students acquired consent from workers, according to our IRB. The worker pool was global and students experimented with different qualification criteria throughout the experiment. In the following sections we analyze these reports to identify common strategies used for data collection tasks. The entire body of data was curated and released as the first free database of meta-information on Computer Science faculty at top graduate programs.

Strategies for Data Collection Tasks

We report six strategies that arose after analyzing the nineteen students' reports. Following grounded theory methodologies (Strauss and Corbin 1990), we classified the students' approaches into strategies based on how the tasks were divided into individual HITs. Figure 1 summarizes the processes that each strategy involves.

Brute-Force (I)

We name the crudest strategy Brute-Force, as it borrows the characteristics and power of brute-force algorithms. In this strategy, students assigned the entire data collection task to a single worker. The worker was requested to fill 10 missing pieces of information for all Computer Science professors of a specific university. Most students employing this strategy used links to external interfaces, such as Google spreadsheets, to more easily accept the full record of faculty. Different styles in the incentives that were used to increase data accuracy were reported. Some students restricted the range of accepted answers by using data validation techniques. A student contemplated using an extreme adaptation of the Brute-Force strategy by creating a single task for the report of all Computer Science professors of all universities. He did not pursue this path due to time and budget constraints; instead all students that used Brute-Force applied it independently to each university. Arguably, Brute-Force includes highly laborious and time consuming tasks for workers, and the quality of the final results relies heavily on the aptitude of a handful of workers. On the other hand, it requires little oversight by the requester, and with the luck of a few dedicated workers, yields rapid and good results.

Column-by-Column (II)

In this strategy, students broke the data collection task into smaller sub-tasks, where each sub-task requests a specific subset of the required information vertically across all faculty. In most observed cases, students employed a worker to create the directory for a specific university, and afterwards created at least three sub-tasks that corresponded to the rank, educational information, and finally research area and join year for all faculty. This strategy can be seen as filling the dataset column-by-column or by groups of similar concepts. Students that used it found that its main advantages are the specialization of each worker on a sub-task, and the isolation of any particularly bad worker's errors to a well-defined subset of the collected data. As a drawback they note its parallelization, as multiple workers will visit the same pages to extract different pieces of information, ignoring adjacent and relevant data on those pages.

Iterative (III)

In this form of crowdsourcing, students asked multiple workers to contribute to the same data collection task, one after another. Filling or correcting a cell in the spreadsheet contributed to a prescribed reward. Each worker could choose to add as many new cells as they wish or edit preexisting data from other workers, as they had access to the whole spreadsheet. Their compensation was based on the extent to which they improved the data over its previous state. This model is mainly characterized by the great potential for interaction between workers; workers may work one after another, with or without temporal overlap, and may interact with each other's results incrementally. Students found that the biggest downside of this strategy is the complexity in calculating the compensation, which requires an evaluation of the quality and amount of work done by each worker. A notable consequence of the flexibility of this strategy is that a single worker may choose to complete all the work, and the strategy converges to Brute-Force.

Classic Micro (IV)

The classic micro strategy is a staple of crowdsourcing. In this approach, students often initially asked a worker to generate an index of names and sources for all professors at a given university, and from there divided the job into as many tasks as there were professors. In other words, each worker was employed to retrieve all information about one specific

[3] http://academic.research.microsoft.com

professor. The use of a large pool of workers is a benefit of this strategy, since it isolates each worker's input to a small subset of the data; this ensures that one worker's tendency to make mistakes does not negatively impact the entirety of the final results. However, the need for a large number of workers makes it more difficult to narrow workers by expertise or other credentials, or to interact personally with workers, while keeping the data collection rapid or efficient.

Chunks (V)

Division by chunks can be characterized as the transpose of the Column-by-Column strategy. In Chunks, after compiling an index of professor names and URLs, the faculty was divided into some number of N disjoint groups, and each of N workers were assigned to one group. Some students did not release tasks that required unique workers, thus the majority of the work was completed by only a few workers. It is worth noting that the same scenario can happen with Classic Micro that then converges on Chunks, even if that was not the intent of the novice requester.

Nano (VI)

In this final strategy students divided micro-tasks into a subentry level. After crowdsourcing an index of names and URLs, hundreds of new tasks were released to workers, each requesting a single piece of information given only a professor name, link to a website, and university name. This strategy had the highest level of parallelism and varied in execution based on how many workers eventually did the work. If a worker can work only on one task then the downside is that they miss any benefit from learning to complete the task more efficiently or visiting a page that contains many pieces of relevant information. Additionally, we observed that this type of task may be more difficult for workers to understand and thus complete effectively, since by barring specific details in the instructions, each worker completely lacks any context for the work they are completing. A student that used Nano reports that workers were reluctant to admit they did not find the requested information and tended to insert random answers. For example, even though most professors have not completed a postdoc, workers would provide unrelated information that they either found in the resumes or just copied from the instructions of the task.

Strategies for Data Verification

Given the time and budget constraints, only about half of the students created separate tasks that verified and increased the accuracy of their data. Most of the students who did not create verification tasks mentioned that they planned a verification step but had to cancel or abort it later due to budgeting issues. In this section we report three verification schemes students used and discuss their merits and disadvantages based on our observations from the reports.

Magnitude

Many students employed one or more workers to make a count of all faculty at a university before issuing any tasks for more specific information, such as faculty names; this measure of a faculty's magnitude serves as a quick and inexpensive sanity check for completeness of information collected later. Additionally, if magnitudes are collected redundantly, the level of agreement between independent workers' reported magnitudes can serve as a reflection on the initial difficulty of a task. In this experiment, for example, many concurring reports of a faculty's magnitude might reflect a department with a well-organized faculty directory.

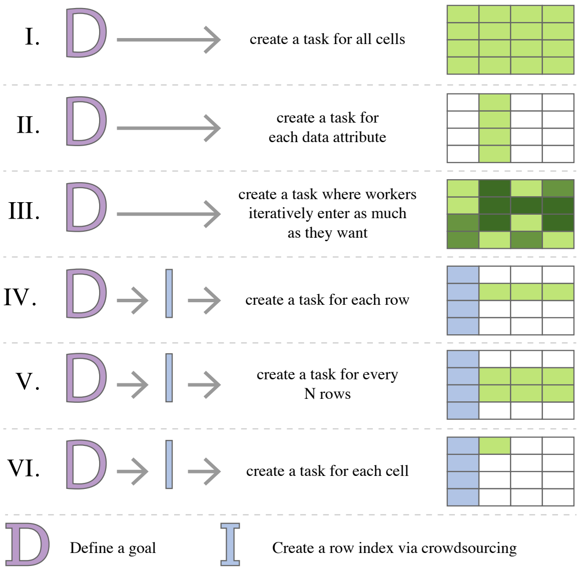
I.
create a task for all cells
II.
create a task for each data attribute
III.
create a task where workers iteratively enter as much as they want
IV.
create a task for each row
V.
create a task for every N rows
VI.
create a task for each cell
Define a goal
Create a row index via crowdsourcing

Figure 1: An illustration showing the process for each of the reported six crowdsourcing strategies. These are: (I) Brute-Force, (II) Column-by-Column, (III) Iterative, (IV) Classic Micro, (V) Chunks, and (VI) Nano.

Redundancy

Some students created redundancy verification tasks, asking a small number of workers to do the same task independently from one another, similar to approaches used in well-known crowdsourcing projects (von Ahn and Dabbish 2004). They came up with different ways of merging those multiple results, including iteratively through a voting scheme, through use of a script, and by simply selecting the best dataset.

Reviewer

The third verification strategy employs a reviewer, an independent worker who inspects and edits all or part of the collected data, similarly to oDesk's hierarchical model. This strategy is notably challenging, since it can be very difficult to automatically validate the reviewer's changes or to ensure that the reviewer did a complete evaluation of the data they were given. We observe that this strategy worked particularly well when used in conjunction with incremental incentives such as rewarding reviewers proportionally to the improvements they make, but simultaneously being cautious of greedy reviewers who attempt to take advantage of the bonus system and introduce incorrect changes. In some cases, it seemed beneficial to employ a meta-reviewer to val-

Dimension	Values
Incentive	monetary bonus, recognition of work, higher standard payments, none
Interface type	native, third-party, custom
Task size	small, medium, large
Level of guidance	example and strategy, strategy only, none
Input restrictions	forced choices, suggested choices, free text
Interaction among workers	restrictive, incremental, none

Table 1: The six dimensions we discovered in the crowdsourcing strategies observed in the class-sourcing activity.

idate the first reviewer's work or parallel reviewers, though this scheme can easily become complex and unmanageable.

Task Dimensions

In this section, we identify and analyze crowdsourcing task design choices made by the novice requesters. To the best of our knowledge, this is the first attempt to understand the choices of novice requesters and provide an analysis of their strategies based on empirical findings, on any particular type of task. Most of the categories we describe can be generalized to many types of tasks, such as image categorization or video annotation. We strive to provide a classification that is independent of the specificities of MTurk.

By performing an open coding of the student-submitted project reports, we identify six distinct dimensions that describe features of the crowdsourcing strategies: incentive, interface type, task size, level of guidance, level of restrictions, and interaction among workers. We describe the six dimensions in detail and summarize our findings in Table 1.

Incentives

The most common incentive that students provided to the workers is monetary bonuses in addition to their normal payments as a reward for exceptional work. These bonuses are often promised by the students and stated in the description of the task to tempt workers into increasing the quality of their work. Some students gave bonuses for single tasks while others grouped tasks and checked the overall activity of their workers. Sometimes workers received an unexpected bonus by a satisfied requester. Another incentive we observed is the interaction that some students initiated with their workers. The class reports suggest that workers tend to be highly appreciative of such non-financial recognition and strive to prove their value to these requesters. Finally, some students did not give any bonus or non-financial incentive but gave a higher standard payment instead. A possible motivation for this style of employment is that giving a bonus requires manual evaluation of the quality of the output.

Interface Type

We observed that students used three different types of interfaces to present their tasks. The most broadly used is the native interface that MTurk provides. This interface can be hard to use for novice requesters, but is guaranteed to work and is accepted by all workers. Students often found themselves restricted as the native interface provides limited functionality for more specialized and complex tasks. In this case they linked their tasks to external interfaces offered by third-parties or they designed themselves. A common example in our experiment was the use of Google spreadsheets as an external database that students knew how to work with and most workers were already familiar with. Only one student attempted to direct his workers to a website he hosted his own infrastructure and found it hard to acquire any results. We speculate that using custom webpages can be tricky, as workers hesitate to trust an unknown interface or are simply not interested in getting familiar with it.

Task Size

We classified the tasks designed by the students as small, medium, or large, depending on the time of completion. Small tasks lasted up to 5 minutes and required little effort by the worker. In our context, a small task would be to acquire information for a single professor. Medium tasks were more complicated and could last up to 30 minutes. For example, finding the full names of all the professors of a given university is a task of medium complexity that involves a certain degree of effort, but is relatively straightforward. Large tasks were more demanding and lasted more than an hour. An example of a large task is to find and record all information for all faculty at one university. In addition to dedication, these tasks may require some level of expertise, and as such tend to be accompanied by large rewards.

Level of Guidance

Similarly, we characterize tasks based on different levels of guidance provided by the students. When there was no guidance, workers were given a task and were free to choose the way they would proceed to tackle it. For example, if workers are simply asked to fill in all information for all Computer Science professors of a certain university, they can choose their own strategy towards that end. Some workers might first find the names of all professors and then proceed to more specific information, while others would build the database row by row, one professor at a time. We note that giving no guidance can be a refreshing and creative opportunity for the workers but can unfortunately lead them to inefficient methodologies or entirely incorrect results. We observed two ways of providing guidance: through strategies, or by the additional use of examples. In strategy-based guidance, workers were provided with tips on where and how to find the required information, e.g. "you may find degree information on the CV". In example-based guidance, workers were provided with a scenario of how a similar task is completed. Some students observed that a drawback of using guided tasks is that they can cultivate a lazy culture, where workers only do what is explicitly suggested and do not take the extra step that could lead to higher quality results.

Level of Restriction

We also distinguished tasks based on their level of restriction. In tasks with no restriction students accepted free text answers and granted full control and freedom to the worker. In more controlled tasks students introduced restrictions in certain forms, e.g. pull-down menus that guided workers to a predefined list of acceptable universities for the degree fields. In between the two, we observed that some students provided a list of suggestions but still allowed workers to enter free text in case they can't decide matches best the information they see. It is worth noting common themes we found in the reports. Free text answers can give a better insight to the quality of work, but involve post-processing to bring them to some acceptable format and therefore are not easily reproduced and generalized. Meanwhile, a number of students reported that even though restricted answers can be easily evaluated, they can be easily "poisoned" by malicious or lazy workers who randomly pick a predefined option. Notably, this can also occur in free text answers where lazy workers merely copy the examples given in the description.

Interaction Among Workers

We observed three distinct patterns of interaction among workers: restrictive, incremental, and none. In restrictive interactions, the input from one worker in an early stage served as a constraint for other workers that would work on later stages. For example, some students first posted tasks requesting an index directory or a count of the faculty. They later used those as inputs, to request complete faculty information, while making sure the number of collected faculty approximately agreed with the initial count. In the incremental interaction setting, the workers could see or modify results that were acquired by others. The iterative strategy always involved incremental interactions. This interaction style also appeared in other strategies when multiple workers entered data into a shared space, e.g. a Google Doc. Finally, workers could also work in isolation, without access to any information from other workers. Students observed that the first two models of interaction promoted a collaboration spirit that resembles offline marketplaces, compared to the third where workers work in a more individualistic base.

Analysis of Strategies and Dimensions

In this section, we analyze the identified strategies and dimensions to assess how task accuracy differs across strategies and choices within individual dimensions. The goal here is to better understand what approaches worked well and not so well for the novice requesters based on task accuracy and possible explanations identified in the reports.

Computing Data Accuracy

The structure of the class assignment assigned two students to each one of the 50 universities. One student did not manage to acquire any data for one university, thus we have excluded it. As part of this analysis, we computed the accuracy of each provided dataset for each one of the 99 instances. Even though we lack the ground truth, we know that the curated dataset that was publicly released reflects it to a great degree. As of today, we have received more than 650 corrections by professors and researchers who have improved the data. We moderated and confirmed each suggested correction or addition. In addition, we heavily revised the dataset ourselves making hundreds of corrections. Using this public dataset we created a script that compared the completeness and accuracy of the provided data, by finding the longest common subsequence of the provided and final entries. We expect that the reported accuracies contain some errors, as many students did not limit workers to provide answers from a predefined list, sometimes yielding numerous names for the same university e.g. University of Illinois, UIUC, University of Illinois at Urbana-Champaign, etc.

Strategies and Data Accuracy

Table 2 shows the distribution of the strategies. Since the assignment was not a controlled experiment and each student was free to come up and experiment with different strategies, the distribution of the used strategies is not uniform.

We performed an one-way ANOVA to test if the choice of strategy had an effect on the accuracy of the data collected. Despite the unequal sample size, Levene's test did not show inequality in variance. The ANOVA result was significant, $F_{(5,93)}$ =2.901, p =0.018. A post hoc Tukey test showed that data collected using the Classic Micro are significantly more accurate than those obtained using either Chunks (p =0.025) or Column-by-Column (p =0.046). Given that the success of a strategy depends on the way that it was carried and implemented by each student we are hesitant to provide strong claims on the observed differences. Instead we provide possible explanations that rely on the anecdotal evidence found in the reports. We speculate that Column-by-Column is not as efficient as its reward is virtually lower, providing smaller incentives for accurate work: workers have to visit multiple webpages instead of finding everything in a single faculty profile. Chunks suffered in the absence of a faculty directory. Lazy workers who were asked to fill all faculty whose surname fell between a range of letters did a superficial job by providing partial data and submitting premature work. In Classic Micro, requesters rely on more workers and can better control the quality. The mean and standard errors of the data accuracy of each strategy are shown in Figure 2.

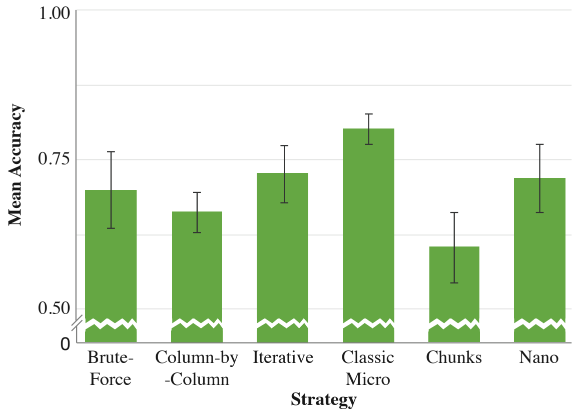

Figure 2: Mean accuracy achieved using each strategy. Error bars show standard errors.

Figure 3 depicts the means of the offered incentives for each strategy. In addition we analyzed the correlation between rewards and quality and completeness of data and

Brute-Force	Column-by-Column	Iterative	Classic Micro	Chunks	Nano
14	23	14	33	10	5

Table 2: The distribution of strategies chosen by students.

found none ($R^2 = 0.016$). Further, the rewards varied greatly. Given the budge constraint of $30 students utilized different techniques to decide how much to pay for each type of task. Some took into consideration the estimated task difficulty over the time required to complete the task, while others mentioned that they tried to match the minimal wage when possible. In cases of accidental errors in the task design, workers were paid more to compensate for their extra effort. Finally, when tasks were completed at a very slow rate students occasionally increased the payment as an attempt to attract more workers. As students lacked mechanisms that informed them of fair and efficient payments, various such techniques were used until their budget was depleted.

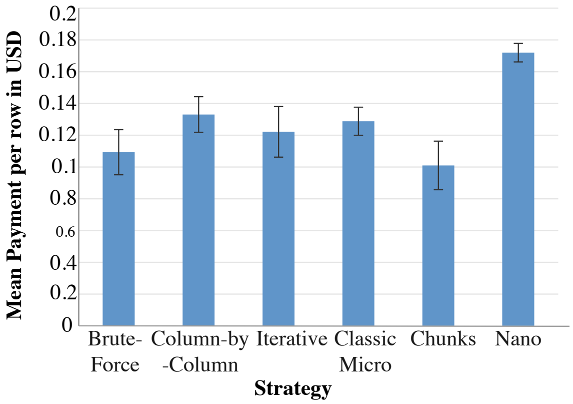

Figure 3: Mean payment per row of data (one faculty entry) for each strategy. Error bars show standard errors.

Dimensions and Data Accuracy

We also analyzed the influence of each dimension's value on data accuracy. Table 3 displays for each dimension the number of times each value is chosen and the corresponding mean accuracy. We note that due to the small sample size and imbalanced choices of dimensions, some of the dimension values were used by only one student (such as giving no incentives), and the accuracies in those cases are inconclusive. We performed a statistical test for each dimension to test whether the choice on any specific dimension influences the data collection accuracy. We performed one-way ANOVAs for all dimensions. For the incentive we performed a Friedman test due to unequal variance among groups.

We found that groups with differing interactions among workers had significant differences in accuracy, $F_{(2,98)}$ =3.294, p =0.04. A post hoc Tukey test showed that data collected from designs where inputs from some workers are used as constraints for others are significantly more accurate than designs with no interactions, p =0.04. Table 3 shows that designs with incremental interactions also on average yield more accurate data than designs with no interactions. This suggests that building worker interactions into a crowd task design, either implicitly as constraints or explicitly as asynchronous collaboration, may improve the quality of the data collected. Indeed, we have identified several ways that worker interactions can reduce task difficulties for individual workers: 1) by providing extra information, e.g. the link to a professor's website collected by another worker who completed an early stage task; 2) by providing examples, e.g. in the incremental interaction setting a worker can see what the information collected by previous workers looks like; 3) by allowing verification during data collection, since workers can edit other workers' results.

Dimension	Value	Times Used	Mean Accuracy
incentive	monetary bonus	49	0.70
	higher standard payment & recognition of work	17	0.74
	monetary bonus & recognition of work	28	0.73
	none	5	0.83
interface	native	57	0.70
	third-party	39	0.75
	custom	3	0.72
size	small	65	0.74
	medium	15	0.65
	large	19	0.72
guidance	example & strategy	56	0.73
	strategy only	39	0.70
	none	4	0.80
input restrictions	forced choices	55	0.73
	suggested choices	4	0.80
	free text	40	0.70
worker interactions	restrictive	42	0.77
	incremental	30	0.70
	none	27	0.66

Table 3: The effect of different values for each dimension. The Times Used column corresponds to the total number of tasks with a specific value for a dimension. The Mean Accuracy column corresponds to the quality of the acquired data for each dimension.

We did not find any significant effect when varying the amount of compensation, which may be surprising to some. However, when we coded monetary bonus, higher standard payment, and recognition into three separate binary dimensions and performed a t-test on each, we found that requesters communicating with workers to show recognition gathered data that are significantly more accurate than those who did not, p =0.021. We did not find an effect of the monetary bonus or the higher standard payment. Acknowledging the good work, or even indicating mistakes seems beneficial. This is consistent with previous findings on the effect of feedback to crowd workers (Dow et al. 2012). We also did not find any significant effect of interface type, task size, level of guidance, and input restrictions.

Guidelines for Novice Requesters

By analyzing the students' strategies and extracting their facets, we are able to tap into how straightforward choices made by novices affect the outcomes of crowdsourcing. We summarize our findings into simple guidelines that can be adopted by novice requesters who are new to using crowdsourcing for data collection and extraction from the Web.

Guideline 1: *Consider the level of guidance and the amount of provided training when choosing the task size, instead of simply choosing the "middle ground".*

During the experiment, strategies with varying task sizes were deployed. We observe that for different task sizes, there is generally a trade-off between worker experience and worker diversity. For larger tasks, each individual worker accomplishes a large amount of work, thus being provided training opportunities. Apart from experience that can be gained just from doing more work, it is also practical and cost-effective for requesters to train a worker by giving feedback; this is especially important for workers that will contribute to a significant amount of work. One risk of having large tasks is that if the task is accepted and completed by a careless worker, then the overall quality of the results may suffer; communication with workers, however, can also help reduce this risk. When a job is posted as a set of smaller tasks, workers with diverse skills will be engaged, and the quality of individual work may have less impact on the quality of the final results. However, workers may be less experienced and it is impractical for the requester to give feedback to all of them. The optimal task size may well be dependent upon the level of guidance and the amount of training required for workers to produce high-quality results. We encourage requesters to experiment with this factor when designing jobs. While it may seem tempting to create medium-sized tasks to strike a balance, our result suggests otherwise. The Chunks strategy, which uses medium-sized tasks, turned out to be one of the worst strategies in this study. Multiple design choices may be contributing to its ineffectiveness, however, the task size may be an important factor: workers may become bored part-way through the task, not remaining motivated by the later larger payoff.

Guideline 2: *Experiment with test-tasks to choose a payment proportional to the task size.*

A task's compensation should be proportional to its size. Requesters ought to be aware of a task's time complexity and should adjust the reward accordingly. Some platforms recommend a minimum hourly wage, and workers seem to expect at least such payment. A few students in the class reported receiving emails from workers complaining the wage was too low and should at least match the U.S. minimum wage. Approximating the allotted time for a specific task can be daunting—requesters tend to underestimate its complexity since they are familiar with the requested information and the processes. Requesters should release test-tasks in order to receive the appropriate feedback from the crowd. It is also essential to experiment with different styles and amounts of incentives in order to achieve the most accurate results.

Guideline 3: *Communicate with workers who have done well in early stages to engage them further.*

We cannot stress enough how important it is to communicate with workers. Throughout the reports we repeatedly observed comments that showed strong correlations between contacting workers and the speed and quality of work. One student not only promised a monetary bonus, but also tried to make the worker part of the project, explaining the time constraint and its scientific value: "While he did realize that he would be getting a good amount of money for his work (and this was a major motivation) he also responded to my high demand of doing a lot of tedious work quickly, because I had told him about my time crunch. Furthermore, task descriptions mentioning that these results will be published also might have played a large role in encouraging the Turkers to do these mundane tasks. Here, human emotions significantly motivate whether one does a task or not!"

Students also sought workers that had a good accuracy and completion record in early stages, to assist them with the rest of the tasks. Most of the workers would respond positively and would accept additional tasks. It is worth noting that some students released tasks with minimum payment of 1 cent to make sure that only their preferred workers would accept them. Later they would fully compensate with bonuses. It is striking how easy it is to form such a bond of trust and we urge requesters of any expertise to explore and take advantage of the effect of the human nature.

In our study, communication might be particularly effective as this task required some domain expertise from workers, such as knowing about subfields in Computer Science. Workers can gain specific knowledge either by doing the tasks or by receiving additional guidance. Thus, it can be more effective to work with experienced workers who have completed similar tasks.

Guideline 4: *Provide additional incentives when using restricted inputs and pair free-text inputs with additional data cleaning tasks.*

When designing the crowdsourcing jobs, requesters decide whether to use free-text inputs or restrict user inputs to a discrete set of options by providing them drop-down lists. Each of the two options has advantages and disadvantages. Providing a drop-down list can eliminate noise in user inputs caused by typos and slight variations in naming conventions. However, they make it easier for "lazy" workers to cheat by randomly choosing an option, and this is hard to detect. Free-text inputs, tend to be more correct despite the need for additional review to remove typos and merge variations. One possible reason for this is that in the case of data collection, free-text inputs ensure that the amount of effort needed to produce rubbish answers that can easily pass the review is as much as that needed to produce good answers, and, as Kittur discussed in (Kittur, Chi, and Suh 2008), there will then be little incentive for workers to cheat. There are methods to make up for the disadvantages of both choices. Restricted inputs can be coupled with bonuses or communication, while noise in free-text inputs can be removed by follow-up verification and cleaning tasks.

Guideline 5: *Plan the data verification methods in advance and carefully consider the amount of effort required for verification.*

One challenge students encountered was the verification of the results, essentially determining whether the data provided by workers were accurate. Many attempted to apply classic verification strategies including peer-reviews and redundancy. Having workers review other workers' output has been shown effective in previous crowdsourcing research (Bernstein et al. 2010). However, students have been generally unsuccessful in this experiment when applying the reviewer strategy: many reported that they would need to pay more for the review task than the data generation task or otherwise no worker would accept the review tasks. A closer look reveals a probable cause: unlike previous applications of reviewer verification where the reviewer only needs to read inputs from other workers and use general knowledge to judge the inputs, in this data collection task the reviewers need to check the information source together with other workers' inputs to judge their correctness. The action of checking the information source potentially requires as much or even more effort as in the data generation task, and is therefore more costly. Compared to reviewer verification, redundancy seems to have been working better in this experiment, though still not as well as for common data collection tasks, since workers are more likely to make the same mistake due to lack of domain expertise. Overall, designing verification strategies for domain-specific data collection tasks seems challenging, and requesters may need to carefully consider the amount of effort required for each action and the influence of domain expertise in this process.

Discussion

Responsibility of Crowdsourcing Platforms

There exist a plethora of tools and systems built both separately from existing crowdsourcing platforms and on top of such platforms for the purpose of improving requesters ranging from novices to experts with a wide variety of general and highly specific crowdsourcing tasks. Given this wealth of research and development into the shortcomings of current platforms and methods of improving them, the onus now falls on those platforms to integrate such community feedback. In particular, this work suggests several reasonable improvements that would improve novice requesters' experiences with crowdsourcing platforms such as MTurk. For instance, given the importance of communication with workers, crowdsourcing platforms should consider providing the functionality and an accessible interface for allowing requesters to chat in real time with their workers, and to assign certain tasks directly to specific workers in the event that a requester prefers that model. Similarly, novice users' difficulty anticipating the costs and best practices for data verification, existing platforms could draw on systems like Soylent and Crowdforge to improve data collection tasks by automatically generating redundant data and making a first algorithmic attempt at merging the results.

Ethics

We notice that a substantial number of students expressed concerns on the fairness of their payment and the use of crowdsourcing in general. A student wrote: "I felt too guilty about the low wages I was paying to feel comfortable [...] I would not attempt crowdsourcing again unless I had the budget to pay workers fairly." A possible explanation for such feelings is the unfamiliarity with crowdsourcing for the majority of the students. Further, the typical reward per task converted to be significantly less than federal required minimum wage in the United States. Being unaccustomed with payment strategies led to poor budget management, which afterwards hindered their ability to pay workers fairly. The idealistic views of 19 students are perhaps not representative of the general feeling of crowdsourcing requesters and might not accurately reflect what the workers perceive to be fair compensation. Nevertheless, this warrants consideration, as there is currently no universal consensus on what is considered to be a reasonable or generous compensation by both requesters and workers in this branch of the free market.

Limitations

Since crowdsourcing platforms do not provide any demographics about requesters, there can be concerns of what constitutes a "typical" novice requester and if students actually fit that definition. As with all crowdsourcing projects, the quality of the data can substantially differ depending on the worker performing the task. Similarly, some students were more careful than others in completing the assignment. Although our students were motivated by grade requirements and personal drive for in-class excellence, novice requesters might be motivated by a variety of different factors.

An additional limitation is that the evaluation comprised of only one data collection task. There are potentially different types of data to collect, and each may present different challenges. We decided to explore one task in depth, and one that was both realistic and had real practical value.

Conclusion

This paper presents the first analysis of crowdsourcing strategies that novice requesters employ based on the following experiment: 19 students with limited experience use crowdsourcing to collect basic educational information for all faculty in 50 top U.S. Computer Science departments. We provide an empirical classification of six specific crowdsourcing strategies which emerged from students' reports. We also compare these strategies based on the accuracy of the data collected. Our findings show that some design choices made by novice requesters are consistent with the best practices recommended in the community, such as communicating with workers. However, many struggled with coming up with cost-effective verification methods.

These findings imply several guidelines for novice requesters, especially those interested in data collection tasks. Requesters, we found, can issue successful data collection tasks at a variety of sizes using workers with a variety of skill levels and classifications. Requesters may also want to carefully consider where and whether they will validate their collected data, since the need for and cost-efficacy of validation can vary greatly between tasks—in some cases, having a worker review the data can cost more than generating the data in the first place. We recommend that requesters interact personally with workers to give clarifications and express appreciation. We also suggest that requesters estimate the size of their tasks before issuing them to workers, and calculate their pay rates accordingly, both for the happiness of their workers and their own consciences.

References

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

Corney, J. R.; Torres-Sánchez, C.; Jagadeesan, A. P.; and Regli, W. C. 2009. Outsourcing labour to the cloud. *International Journal of Innovation and Sustainable Development* 4(4):294–313.

Doan, A.; Ramakrishnan, R.; and Halevy, A. Y. 2011. Crowdsourcing systems on the world-wide web. *Communications of the ACM* 54(4):86–96.

Dow, S.; Kulkarni, A.; Klemmer, S.; and Hartmann, B. 2012. Shepherding the crowd yields better work. In *Proceedings of CSCW*, 1013–1022.

Faradani, S.; Hartmann, B.; and Ipeirotis, P. G. 2011. What's the right price? pricing tasks for finishing on time. *Human Computation* 11.

Franklin, M. J.; Kossmann, D.; Kraska, T.; Ramesh, S.; and Xin, R. 2011. Crowddb: answering queries with crowdsourcing. In *Proceedings of SIGMOD*, 61–72.

Geiger, D.; Seedorf, S.; Schulze, T.; Nickerson, R. C.; and Schader, M. 2011. Managing the crowd: Towards a taxonomy of crowdsourcing processes. In *Proceedings of AMCIS*.

Gutheim, P., and Hartmann, B. 2012. Fantasktic: Improving quality of results for novice crowdsourcing users. Master's thesis, EECS Department, University of California, Berkeley.

Hetmank, L. 2013. Components and functions of crowdsourcing systems - a systematic literature review. In *Wirtschaftsinformatik*, 4.

Huang, E.; Zhang, H.; Parkes, D. C.; Gajos, K. Z.; and Chen, Y. 2010. Toward automatic task design: A progress report. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, 77–85.

Kaufmann, N.; Schulze, T.; and Veit, D. 2011. More than fun and money. worker motivation in crowdsourcing-a study on mechanical turk. In *AMCIS*, volume 11, 1–11.

Kittur, A.; Smus, B.; Khamkar, S.; and Kraut, R. E. 2011. Crowdforge: Crowdsourcing complex work. In *Proceedings of UIST*, 43–52.

Kittur, A.; Chi, E. H.; and Suh, B. 2008. Crowdsourcing user studies with mechanical turk. In *Proceedings of CHI*, 453–456.

Kulkarni, A.; Can, M.; and Hartmann, B. 2012. Collaboratively crowdsourcing workflows with turkomatic. In *Proceedings of CSCW*, 1003–1012. ACM.

Lasecki, W. S.; Miller, C. D.; Kushalnagar, R.; and Bigham, J. P. 2013. Real-time captioning by non-experts with legion scribe. In *Proceedings of ASSETS*, 56:1–56:2.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2010. Turkit: human computation algorithms on mechanical turk. In *Proceedings of UIST*, 57–66. ACM.

Malone, T. W.; Laubacher, R.; and Dellarocas, C. 2010. The collective intelligence genome. *IEEE Engineering Management Review* 38(3):38.

Marcus, A.; Wu, E.; Karger, D.; Madden, S.; and Miller, R. 2011. Human-powered sorts and joins. *Proceedings of the VLDB Endowment* 5(1):13–24.

Quinn, A. J., and Bederson, B. B. 2011. Human computation: a survey and taxonomy of a growing field. In *Proceedings of CHI*, 1403–1412.

Strauss, A., and Corbin, J. 1990. *Basics of Qualitative Research*. Sage Publications, Inc.

von Ahn, L., and Dabbish, L. 2004. Labeling images with a computer game. In *Proceedings of CHI*, 319–326.

Willett, W.; Heer, J.; and Agrawala, M. 2012. Strategies for crowdsourcing social data analysis. In *Proceedings of CHI*, 227–236.