Understanding Workers, Developing Effective Tasks, and Enhancing Marketplace Dynamics:

A Study of a Large Crowdsourcing Marketplace

Ayush Jain
ajain42@illinois.edu
University of Illinois

Akash Das Sarma
akashds@cs.stanford.edu
Stanford University

Aditya Parameswaran
adityagp@illinois.edu
University of Illinois

Jennifer Widom
widom@stanford.edu
Stanford University

ABSTRACT

We conduct an experimental analysis of a dataset comprising over 27 million microtasks performed by over 70,000 workers issued to a large crowdsourcing marketplace between 2012-2016. Using this data—never before analyzed in an academic context—we shed light on three crucial aspects of crowdsourcing: (1) Task design — helping requesters understand what constitutes an effective task, and how to go about designing one; (2) Marketplace dynamics — helping marketplace administrators and designers understand the interaction between tasks and workers, and the corresponding marketplace load; and (3) Worker behavior — understanding worker attention spans, lifetimes, and general behavior, for the improvement of the crowdsourcing ecosystem as a whole.

1. INTRODUCTION

Despite the excitement surrounding artificial intelligence and the ubiquitous need for large volumes of manually labeled training data, the past few years have been a relatively tumultuous period for the crowdsourcing industry. There has been a recent spate of *mergers*, e.g., [21], *rebrandings*, e.g., [16], *slowdowns*, e.g., [15], and moves towards *private crowds* [29]. For the future of crowdsourcing marketplaces, it is therefore both important and timely to step back and study how these marketplaces are performing, how the requesters are making and can make best use of these marketplaces, and how workers are participating in these marketplaces—in order to *develop more efficient marketplaces, understand the workers' viewpoints and make their experience less tedious, and design more effective tasks from the requester standpoint.*

At the same time, developing a better understanding of how crowdsourcing marketplaces function can help us design crowdsourced data processing algorithms and systems that are more efficient, in terms of latency, cost, and quality. Indeed, crowdsourced data processing is performed at scale at many tech companies, with tens of millions of dollars spent every year [29], so the efficiency improvements can lead to substantial savings for these companies. In this vein, there have been a number of papers on both optimized algorithms, e.g., [19, 32, 13, 36, 8, 12, 34], and systems, e.g., [28, 17, 27, 9, 33], all from the database community, and such findings can have an impact in the design of all of these algorithms and systems.

Proceedings of the VLDB Endowment, Vol. 10, No. 7
Copyright 2017 VLDB Endowment 2150-8097/17/03.

Unfortunately, due to the proprietary nature of crowdsourcing marketplace data, it is hard for academics to perform such analyses and identify pain points and solutions. Fortunately for us, one of the crowdsourcing marketplaces made a substantial portion of its data from 2012 to date available to us: this includes data ranging from worker answers to specific questions and their respective response times, all the way to the HTML that encodes their user interfaces.

This data allows us to answer some of the most important open questions in microtask crowdsourcing: what constitutes an "effective" task, how can we improve marketplaces, and how can we enhance workers' interactions. In this paper, using this data, we study the following key questions:

- Marketplace dynamics: helping marketplace administrators understand the interaction between tasks and workers, and the corresponding marketplace load; e.g., questions like: (a) how much does the load on the marketplace vary over time, and is there a mismatch between the number of workers and the number of tasks available, (b) what is the typical frequency and distribution of tasks that are repeatedly issued, (c) what types of tasks and data are requesters most interested in?
- Task design: helping requesters understand what constitutes an effective task, and how to go about designing one; e.g., questions like: what factors impact (a) the accuracy of the responses; (b) the time taken for the task to be completed; or (c) the time taken for the task to be picked up? Do examples and images help? Does the length or complexity of the task hurt?
- Worker behavior: understanding worker attention spans, lifetimes, and general behavior; e.g., questions like (a) where do workers come from, (b) do workers from different sources show different characteristics, such as accuracies and response times, (c) how engaged are the workers within the marketplace, and relative to each other, and (d) how do their workloads vary?

The only paper that has performed an extensive analysis of crowdsourcing marketplace data is the recent paper by Difallah et al. [14]. This paper analyzed the data obtained via crawling a public crowdsourcing marketplace (in this case Mechanical Turk). Unfortunately, this publicly visible data provides a restricted view of how the marketplace is functioning, since the worker responses, demographics and characteristics of the workers, and the speed at which these responses are provided are all unavailable. As a result, unlike the present paper, that paper only considers a restricted aspect of crowdsourcing marketplaces, specifically, the price dynamics of the marketplace (indeed, their title reflects this as well)—for instance, demand and supply modeling, developing models for predicting throughput, and analyzing the topics and countries preferred by requesters. Even for marketplace dynamics, to fully distinguish the results of the present paper from that paper, we exclude any experiments or analyses that overlap with the experiments performed in that paper. We describe this and other related work in Section 6.

This experiments and analysis paper is organized as follows:

- **Dataset description and enrichment.** In Section 2, we describe what our dataset consists of, and the high-level goals of our analysis. In Section 2.1 through 2.3 we provide more details about the marketplace mechanics, the scale and timespan of the dataset, and the attributes provided. We also enrich the dataset by manually labeling tasks ourselves on various features of interest, described in Section 2.4, e.g., what type of data does the task operate on, what sort of input mechanism does the task use to get opinions from workers.
- **Marketplace insights.** In Section 3, we address questions on the (a) marketplace load — *task arrivals* (Section 3.1), *worker availability* (Section 3.2), and *task distribution* (in our technical report [24]), with the aim of helping improve marketplace design, and (b) the types of tasks, *goals, human operators and data types*, and correlations between them (Section 3.3), with the aim of characterizing the spectrum of crowd work.
- **Task design improvements.** In Section 4, we (a) characterize and quantify metrics governing the "effectiveness" of tasks (Section 4.1), (b) identify features affecting task effectiveness and detail how they influence the different metrics (Sections 4.3 through 4.7), (c) perform a classification analysis in Section 4.9 wherein we predict the various effectiveness metric values of a task based on simple features, and (d) provide summarized recommendations on how requesters can improve their tasks' designs to optimize for these metrics (Section 4.8).
- **Worker understanding.** In Section 5, we analyze and provide insights into the worker behavior. We compare characteristics of different worker demographics and sources—provided by different crowdsourcing marketplaces; as we will find, the specific marketplace whose data we work with solicits workers from many sources (Section 5.1). We also provide insights into worker involvement and task loads taken on by workers (Section 5.2), and characterize worker *engagement* (Section 5.3).

2. DATASET DETAILS

We now introduce some terms that will help us operate in a marketplace-agnostic manner. The unit of work undertaken by a single worker is defined to be a *task*. A task is typically listed in its entirety on one webpage, and may contain multiple short *questions*. For example, a task may involve flagging whether each image in a set of ten images is inappropriate; so this task contains ten questions. Each task operates on a set of *items*; in our example, each image is an item. Tasks are issued by *requesters*. Often, requesters issue multiple tasks in parallel so that they can be attempted by different workers at once. We call this set of tasks a *batch*. Requesters often use multiple batches to issue identical units of work—for example, a requester may issue a batch of 100 "image flagging" tasks one day, operating on a set of items, and then another batch of 500 "image flagging" tasks after a week, on a different set of items. We overload the term *task* to also refer to these identical units of work issued across time and batches, independent of the individual items being operated upon, in addition to a single instance of work. The usage of the term task will be clear from the context; if it is not clear, we will refer to the latter as a *task instance*.

2.1 Operational Details

Due to confidentiality and intellectual property reasons, we are required to preserve the anonymity of the commercial crowdsourcing marketplace we operate on, who have nevertheless been generous enough to provide access to their data for research purposes. To offset the lack of transparency due to the anonymity, we discuss some of the crucial operational aspects of the marketplace, that will allow us to understand how the marketplace functions, and generalize from these insights to other similar marketplaces.

The marketplace we operate on acts as an aggregator or an intermediary for many different sources of crowd labor. For example, this marketplace uses Mechanical Turk [3], Clixsense [1], and NeoDev [4], all as sources of workers, as well as an internal worker pool. For task assignment, i.e., assigning tasks to workers, the marketplace makes use of both push and pull mechanisms. The typical setting is via *pull*, where the workers can choose and complete tasks that interest them. In a some sources of workers that we will discuss later on, tasks are *push*ed to workers by the marketplace. For example, Clixsense injects paid surveys into webpages so that individuals browsing are attracted to and work on specific tasks. In either case, the marketplace allows requesters to specify various parameters, such as a minimum accuracy for workers who are allowed to work on the given tasks, any geographic constraints, any constraints on the sources of crowd labor, the minimum amount of time that a worker must spend on the task, the maximum number of tasks in a batch a given worker can attempt, and an answer distribution threshold (i.e., the threshold of skew on the answers provided by the workers below which a worker is no longer allowed to work on tasks from the requester). The marketplace monitors these parameters and prevents workers from working on specific tasks if they no longer meet the desired criteria. We provide additional details of our data collection process in our technical report.

2.2 Origin of the Dataset

Our dataset consists of tasks issued on the marketplace from 2012–16. Unfortunately, we do not have access to all data about all tasks. There are about 58,000 batches in total, of which we have access to complete data for a sample of about 12,000 batches, and minimal data about the remaining, consisting of the title of the task and the creation date. Almost 51,000, or 88% of the 58,000 of batches have some representatives in our 12,000 batch sample—thus, the sample is missing about 10% of the tasks. (That is, there are identical tasks in the 12,000 batch sample.) From the task perspective, there are about 6600 distinct tasks in total, spread across 58,000 batches, of which our sample contains 5000, i.e., 76% of all distinct tasks. Thus, while not complete, our sample is a significant and representative portion of the entire dataset of tasks. *We will largely operate on this 12,000 batch sample, consisting of 27M task instances, a substantial number.* Figure 1 compares the number of distinct tasks sampled versus the total number issued to the marketplace across different weeks, We observe that in general we have a significant fraction of tasks from each week.

2.3 Dataset Attributes

The dataset is provided to us at the batch level. For each batch in our sample, we have metadata containing a short textual description of the batch (typically one sentence), as well as the source HTML code to one sample task instance in the batch. In addition, the marketplace also provides a comprehensive set of metadata for each task instance within the batch, containing:

- Worker attributes such as `worker ID`, location (`country`, `region`, `city`, `IP`), and `source` (recall that this marketplace recruits workers from different sources);

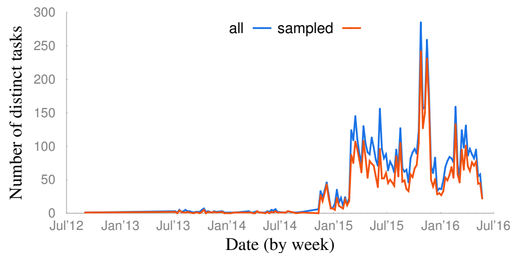


Figure 1: Number of tasks sampled (by week)

- Item attributes such as `item ID`; and
- Task instance attributes such as `task ID`, `start time`, `end time`, `trust score`, and `worker response`.

As we can see in this list, the marketplace assigns workers a `trust score` for every task instance that they work on. This trust score reflects the accuracy of these workers on test tasks the answers to whose questions is known. The marketplace administers these test tasks before workers begin working on "real" tasks.

At the same time there are some important attributes that are not visible to us from this dataset. For instance, we do not have requester IDs, but we can use the sample task HTML to *infer* whether two separate batches have the same type of task, and therefore were probably issued by a single requester. Nor do we have "ground truth" answers for questions answered by workers. However, as we will describe subsequently, we find other proxies to be able to estimate the accuracies of workers or tasks. Finally, we do not have data regarding the payments associated with different tasks.

2.4 How did we enrich the data?

The raw data available for each batch, as described above, is by itself quite useful in exploring high-level marketplace statistics such as the number of tasks and workers over time, the geographic distribution of workers, typical task durations, and worker lifetimes and attention spans. To augment this data even further, we *enrich* the dataset by inferring or collecting additional data. We generate three additional types of task attribute data:

- Manual labels—we also manually annotate each batch using their task interface, on the basis of their *task goal*, e.g., entity resolution, sentiment analysis, *operator type*, e.g., rating, sorting, labeling, and *data type*, e.g., text, image, social media, discussed further in Section 3.3.
- Design parameters—we extract and store *features* from the sample HTML source as well as other raw attributes of the tasks that reflect design decisions made by the requesters. For example, we check whether a task contains instructions, examples, text-boxes and images—we discuss these further in Section 4.
- Performance metrics—we compute and store different metrics to characterize the latency, cost and error of tasks to help us perform quantitative analyses on the "effectiveness" of a task's design, discussed further in Section 4.1.

3. MARKETPLACE ANALYSES

In this section, we aim to gain insights into the high level, aggregate workings of the marketplace. First, we examine some basic statistics of the marketplace, to understand the worker supply and task demand interactions. Specifically, we look at (a) task instance arrival distribution (Section 3.1), (b) worker availability (Section 3.2), and, in our extended technical report [24], we additionally examine the contribution of "heavy-hitter" tasks relative to other tasks that are "one-off" in our marketplace. Then, in Section 3.3, we explore the types of tasks observed in our dataset, to better understand the questions and data types of interest for requesters. We also look for correlations across these labels to understand what types of tasks occur together. In our technical report [24], we additionally explore the complexity of tasks issued over time, and argue that the fraction of "easy" tasks has gone down while the fraction of "hard" ones has gone up.

3.1 Are tasks uniform or bursty over time?

We first study the rate at which task instances arrive into the marketplace, and the rate at which they are completed. We plot the number of task instances arriving and being completed each week in Figure 2a in blue. First, note that the task arrival plot is relatively sparse until Jan 2015, which is presumably when the marketplace started attracting more requesters. Second, after June 2014, there are some very prominent peaks, on top of regular activity each week. This suggests that while task instances arrive fairly regularly, there are periods of *burstiness*. Considering the period from Jan 2015 onwards, the median of the number of task instances issued in a day on the marketplace is about 30,000. In comparison, on its busiest day, more than 900,000 task instances were issued, a 30$\times$ increase over normal levels. Similarly, the number of task instances issued on the lightest day is 0.0004$\times$ of the median. This raises the question: where does the high variation in the number of instances come from—is it a result of fluctuations in the number of batches of tasks issued, or fluctuations in number of distinct tasks themselves? For this analysis, we overlay the number of instances issued on the marketplace with the number of batches and the number of distinct tasks for the period post January 2015 in Figure 2b. For both these measures, we find that their fluctuation is similar to the fluctuation in the number of issued instances, indicating that both factors contribute to the high variation in the market load.

Besides the bursty nature of task instance arrivals across weeks, the marketplace also witnesses periods of low task arrivals on the weekends—the number of instances posted on a weekday is up to 2$\times$ the number of instances posted on Saturdays or Sundays on average. Further, the average number of instances posted at the start of the week is the highest, following which the number decreases over the week. This chart can be found in our technical report [24].

3.2 How does the availability and participation of workers vary?

Worker Availability. As described earlier, the marketplace we work with attracts workers from a collection of labor sources. We investigate the sources the marketplace draws from in Section 5. In this section, we focus on studying the number of active workers across different weeks: Figure 3 depicts this statistic.

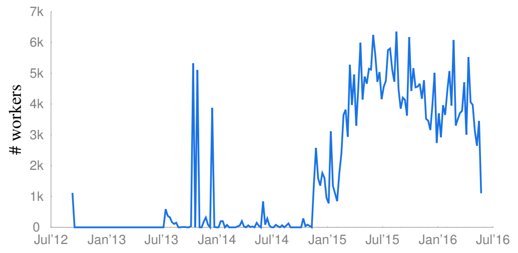
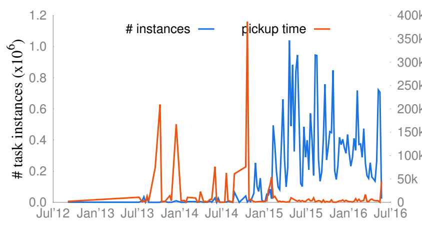
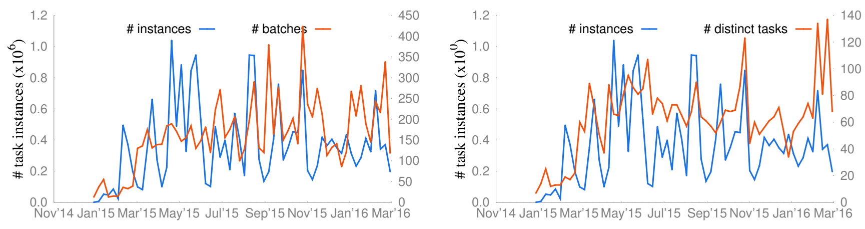


Figure 3: Number of workers performing tasks

Unlike Figure 2a that had a huge variation in the number issued task instances, especially after 2015, Figure 3 does not show this level of variation. Thus, somewhat surprisingly, even though there are huge changes in the number of available task instances, roughly



(a) Task instance arrival vs Median pickup times



(b) Task instance arrival (post Jan 2015) vs (1) batch arrival, (2) distinct task arrival

Figure 2: Task Arrivals by week

the same number of workers are able to "service" a greater number of requests. This indicates that there is a limitation more in the supply of task instances than availability of workers.

Worker Latencies, Idleness, and Task-Distribution. We now attempt to explain how roughly the same number of workers are able to accommodate the variation in the number of tasks on the platform. Our first observation is that the median latency in task instances getting picked up by workers, noted as *pickup time* (and defined formally later in Section 4.1) in Figure 2a, and depicted in red, shows that during periods of high load, the marketplace tends to move faster. We also zoom in to the high activity period after January 2015 in Figure 4a to further highlight this trend. One possible explanation for this observation is that when more task instances are available, a larger number of workers are attracted to the marketplace or recruited via a push-mechanism—leading to lower latencies. Another possibility (supported by our discussion below for Figure 4b) is that with a higher availability of tasks, workers are spending a lot more active time on the platform, and hence are more likely to pick up new tasks as soon as they are available.

Next, we look into how the workload is being distributed across the worker-pool. In Figure 4b, we plot the number of tasks completed by the top-10% (in red color) and the bottom-90% (in green color) of workers in each week and compare it to the total number of tasks issued. We observe that while the bottom-90% also take on a lot more tasks during periods of high load, it is the top-10% that handles most of the flux, and consistently performs a lot more tasks than the remaining 90%. Similarly, examining the same plot for average amount of *active time* spent by workers on the platform in Figure 4b also shows that the top-10% are indeed spending a lot more time on average per week to handle the varying task load as compared to the bottom-90%. This observation indicates that while having a large workforce certainly helps, it is crucial to focus on worker interest and engagement—attracting more "active" workers can allow marketplaces to handle fluctuating workloads better. We also examined the workload handled by workers from different labor sources to verify whether the majority of this variation is assigned to the marketplace's internal or external workers. We observed that the internal workers account for a very small fraction of tasks—we defer a full account of our results to [24].

3.3 What kinds of tasks do we see?

We now study our enriched task-labels from Section 2.4 in order to characterize the spectrum of crowd work in the marketplace. Such an analysis can be very useful, for example, to develop a workload of crowdsourcing, and to better understand the task types that are most important for further research.

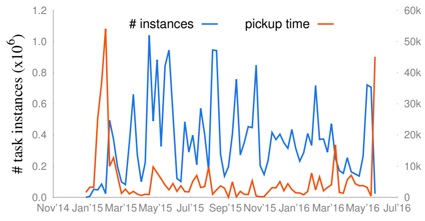
Label Categories. Our mechanism to label tasks is to first cluster batches together based on similarity of constituent tasks, and then label one representative task from each cluster. Since all tasks within each cluster have identical characteristics, we can propagate the labels from the representative task to the rest of the cluster. The goal of our clustering is to capture the separation between distinct tasks, which is not given to us. As labeling is a labor-intensive process, we currently have labels available for about 10,000 out of the total 12,000 batches ($\approx 83\%$) and 24 million out of the total 27 million task instances ($\approx 89\%$). These batches fall into about $\sim$3,200 clusters. We label each task under four broad categories[1]. Tasks have one or more label under each category.

- Task Goal: Here, we separate tasks based on their end goal. We find that most tasks can be characterized as having one (or more) of the following 7 goals[2]: (1) Entity Resolution (ER), for instance, identifying if two webpages talk about the same business, or if two social media profiles correspond to one single person, (2) Human Behavior (HB), including psychology studies, surveys and demographics, and identifying political leanings, (3) Search Relevance Estimation (SR), (4) Quality Assurance (QA), including spam identification, content moderation, and data cleaning, (5) Sentiment Analysis (SA), (6) Language Understanding (LU), including parsing, NLP, and extracting grammatical elements, and (7) Transcription (T), including captions for audio and video, and extracting structured information from images.
- Task Operator: In this category, we label tasks based on the human-operators, or underlying data processing building blocks used by requesters to achieve tasks' goals. We observe primarily 10 different operators: (1) Filter (Filt), i.e., classify items or answer boolean questions, (2) Rate (Rate), i.e., rate items on an ordinal scale (3) Sort (Sort), (4) Count, (5) Label or Tag (Tag), (6) Gather (Gat), i.e., provide information that isn't directly present in the data, for instance by searching the web, (7) Extract (Ext), i.e., convert information implicitly present in provided data into another form, such as extracting text within an image. (8) Generate (Gen), i.e., generate additional information by using inferences drawn from given data using worker judgement and intelligence, such as writing captions for images, (9) Localize (Loc), i.e., identify or mark, and perform actions on specific segments of given data, e.g., draw bounding boxes to identify objects in images, and (10) External Link (Exter), i.e., visit an external webpage and perform an action there, e.g., fill out a survey form, or play a game.
- Data Type: We also separate tasks based on the type of data that is used. The same goals and operators can be applied on multiple data types. All tasks contain a combination of the following 7 types of data: (1) Text, (2) Image, (3) Audio, (4) Video, (5) Maps, (6) Social Media, and (7) Webpage.

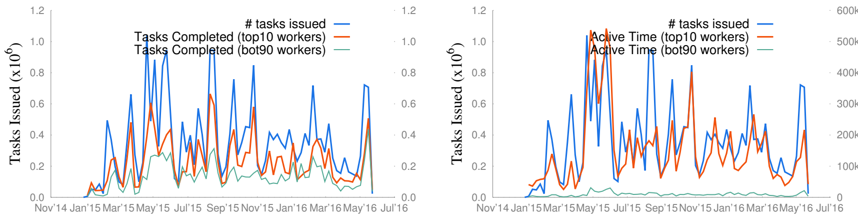
Label distribution. First, we analyze the distribution of labels in different categories across tasks. Figure 5a depicts the popular

[1]Labeling was performed independently by two of the authors, following which the differences were resolved via discussion.

[2]Tasks that had uncommon or unclear goals and did not fall into one of these classes, were automatically classified as Other or Unsure respectively. This holds for the other categories besides goals as well.



(a) Task Arrivals vs Median Pickup Time



(b) Engagement of *top-10%* and *bot-90%* workers: (1) # Tasks (2) Active Time

Figure 4: Task Arrivals by Week (Post Jan 2015)

goals. We observe that complex unstructured data understanding based goals—language understanding and transcription are very common, comprising of over 4 and 3 million tasks, that is around 17% and 13% respectively, despite not having seen extensive optimization research, as opposed to traditional, simpler goals like entity resolution and sentiment analysis that have been extensively analyzed. Figure 5b shows that text and image are still the main types of data available and analyzed — 9.6 million (40%) and 6.3 million (26%) tasks contained text and image data respectively. Audio and video data are also used, and other richer types of data like social media, web pages, and maps are gaining popularity. Figure 5c shows the common operators used. While the distribution of goals indicates that a significant fraction of tasks have complex goals, the underlying operators are still predominantly simple. The marketplace is dominated by the fundamental filter and rate operations — over 8 million (33%) tasks employ some filtering operator, and nearly 3 million (13% of) tasks make use of rating operators. Among more complex operators, we see that gathering, extraction, localization, and generation are frequently applied, together being used in around 5.3 million, i.e., 22% of all tasks.

Goals, operators and data types that occur frequently together. Next, we look at the correlations between the three types of labels for tasks. For example, one question we aim to answer is what kinds of operators are typically applied to different types of data, or used to achieve particular goals? Looking at such correlations across goals, operators, and data types provides fine-grained insights into the structure and design of tasks that is not immediate from our aggregate statistics alone. Here we present three charts in Figure 6 that depict the correlation between each pair; the remaining three charts, along with detailed insights, can be found in the technical report [24]. For instance, Figure 6b shows the breakdown for each goal by the percentage usage of different operators towards achieving that goal; Figure 5c serves as a legend for the stacked bars. We observe that filter and rate operators are used in most kinds of tasks, as well as form a significant majority as the constituent building block for most goals. One notable exception is transcription (which, recall, constitutes over 13% of all tasks by itself, making it a significant exception), where the primary operation employed is extraction. As another example, Figure 6a shows that text and images are important for all types of task goals, for certain types, e.g., ER, SA, SR, social media is also quite important. Lastly, Figure 6c shows that beyond filtering and rating being important, extraction is used quite prominently on text and image data, often rivaling the importance of filtering.

4. EFFECTIVE TASK DESIGN

In this section, we address the question of effective task design. Specifically, we (1) characterize and quantify what constitutes an "effective" task, (2) make data-driven recommendations on how requesters can design effective tasks, and (3) predict the "effectiveness" of tasks based on our hypotheses.

4.1 Metrics for effective tasks

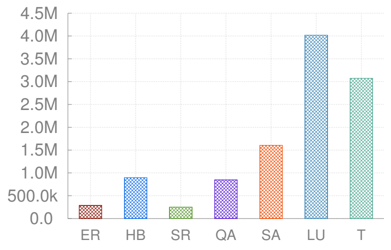
The standard three metrics that are used to measure crowdsourcing effectiveness are: error, cost, and latency. There are various ways these three metrics could be measured; we describe our notions below, given what we can calculate.

Error: Disagreement Score. In our dataset, we have every worker answer provided for each question within each task instance, operating on one distinct item, but not the corresponding ground truth answers. We use these answers to quantify how "confusing" or ambiguous a task is, overall. The way we quantify this is to consider the worker answers for a given question on a given item. If the workers disagree on a specific question on a specific item, then the task is likely to be ambiguous—indicating that it is poorly designed, or hard to answer—either way, this information is important to dictate the task design (e.g., clarify instructions) and the level of redundancy (e.g., more redundancy for confusing questions) that should be adopted by requesters. Our proxy for error is the *average disagreement in the answers for questions on the same item*, across all questions and items in a batch. We consider all pairs of workers who have operated on the same item, and check if their answers are the same or different, giving a score of one if they disagree, and zero if they agree; we then compute the average disagreement score of an item by averaging all these scores; and lastly, we compute the average disagreement score for a batch by averaging the scores across items and questions. We shall henceforth refer to the "Disagreement Score" as disagreement.

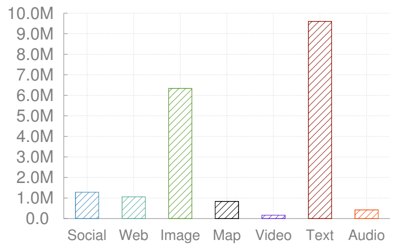
There is however, one small wrinkle. Some operators, and corresponding worker responses may involve textual input. Two textual responses may be unequal even if they are only slightly different from each other. Since textual responses occupy a large fraction of our dataset, it is not possible to ignore them altogether. (We consider this and other strategies in our technical report [24].) We instead adopt a simple rule: we prune away all tasks with disagreement > 0.5 so as to eliminate tasks with very high variations in worker responses. This eliminates the subjective textual tasks, while still retaining the textual tasks that are objective.

Cost: Median Task Time. A typical measure for how much effort a worker has put into a task is the amount of time taken to complete it. Since we do not have information about the actual payments made to workers, we use the median time taken (*in seconds*) by workers to complete tasks in a batch as a proxy for the cost of the batch. This can be calculated from the data that is available, given that we have the start and end times for each task in a batch. We shall subsequently denote the "Median Task Time" by task-time.

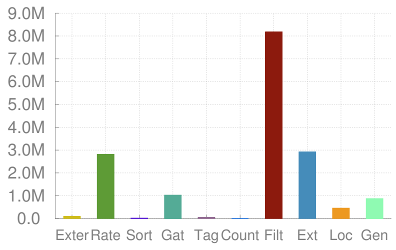
Latency: Median Pickup Time. To characterize latency, we use pickup time, i.e., how quickly tasks are picked up by workers, on average. Pickup time for a batch is computed as follows: pickup-time $=$ median($<$ start time of task$_i$ $-$ batch start time $>$) (*in seconds*). Here, we use the start time of the earliest batch, i.e. start time of task$_1$, as a proxy for the batch start time. We justify this choice for the latency metric quantitatively in our technical report [24]. In short,



(a) Popular Task Goals

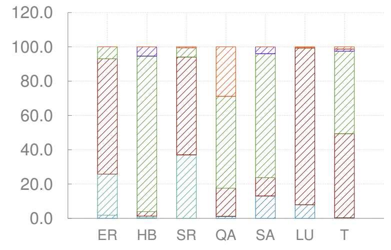


(b) Popular data types

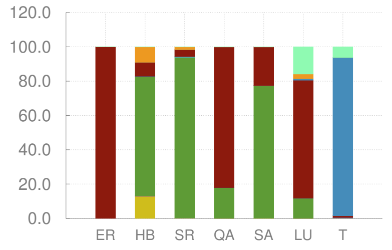


(c) Popular operators

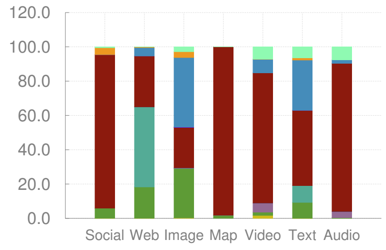
Figure 5: Distribution of goals, data types, and operators



(a) Data used for different task goals



(b) Popular Operators for different task goals



(c) Popular Operators for different data

Figure 6: Correlations across data and goal, operator and goal, and operator and data

we observe that in general the pickup-time for batches is orders of magnitude higher than the task-time, indicating that the latency or total turnaround time of a task is in fact dictated by the rate at which workers accept and start the task instances. We denote the "Median Pickup Time" by pickup-time.

4.2 Correlation Analysis Methodology

In the next set of subsections, we examine some influential features or parameters that a requester can tune, to help improve a task's error (disagreement), cost (task-time) and latency (pickup-time). For instance, features of a task include the length of the task, or the number of examples within it. For each feature, we look at the correlation between the feature and each of the three metrics. We perform a series of (correlation-investigating) experiments, each of which corresponds to one {feature, metric} pair. All our experiments follow the following structure:

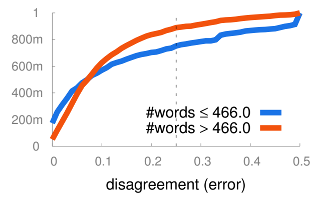
- **Cluster:** We first cluster batches based on the task in order to not have the "heavy-hitter" tasks that appear frequently in multiple batches across the dataset to dominate and bias our findings. Since our analysis will also involve *matching*, or clustering tasks further based on labels, we restrict our focus to the set of around 3,200 labeled clusters corresponding to 83% of all batches and 89% of all task instances. Subsequently, for each cluster, we take the median of metric values across batches, as well as the median of the feature being investigated.
- **Binning:** We separate the clusters into two bins based on their feature value — all clusters with feature value lower than the global median feature value go into Bin-1 (say), while the ones with feature value higher than the median go into Bin-2. (Clusters with feature value exactly equal to the median are all put into either Bin-1 or Bin-2 while keeping the bins as balanced as possible.) For each metric, we examine it's value distribution in the two bins — in particular, we look for differences between the average, median, or distribution of metric values in the two bins. A significant difference indicates a correlation between the feature we have binned on, and the metric being looked at.
- **Statistical significance:** We perform a *t-test* to check whether the metric value difference in our two feature-separated bins is statistically significant. We use a threshold *p-value* of 0.01 to determine significance, i.e., we reject the null hypothesis (that bins have similar metric values) if the p-value is less than 1%.
- **Visualization:** For each feature-metric pair, we plot a cumulative distribution (CDF) plot, with the metric value plotted along the x-axis. Each of the two bins corresponds to one line in the plot. For $x = m$, the corresponding y value on each of the lines represents the probability that a batch will have metric value better than m. Thus, a higher value is preferable; and we compare the two bins (or lines) in this plot.

In Sections 4.3-4.7, we look at the results for some of the significant correlations we found. In our technical report [24], we further support our claims from these sections by providing examples and comparing qualities of real tasks issued on the marketplace that are dissimilar in individual features but similar in other respects.

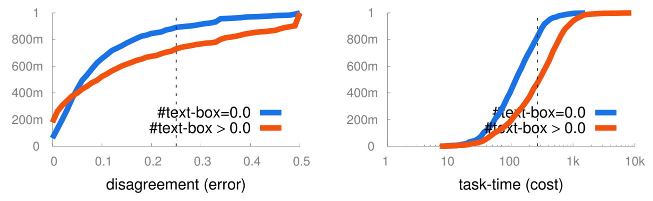
4.3 Number of HTML words

We examine how the length of task—defined as the number of words in the HTML page, and denoted as #words—impacts the effectiveness of the task. We show the effect of length of task on our metrics in Figure 7a. We observe that the line for clusters with higher #words in their HTML interface dominates, or is above the line for the clusters with fewer #words. This may be because longer tasks tend to be more descriptive, and the detailed instructions help reduce ambiguity in tasks, train workers better, and thereby reduce mutual disagreement in answers. We also note that the length of the task does not significantly affect either the pickup-time or task-time metrics. Thus, workers are neither discouraged nor slowed down by longer textual descriptions of tasks.

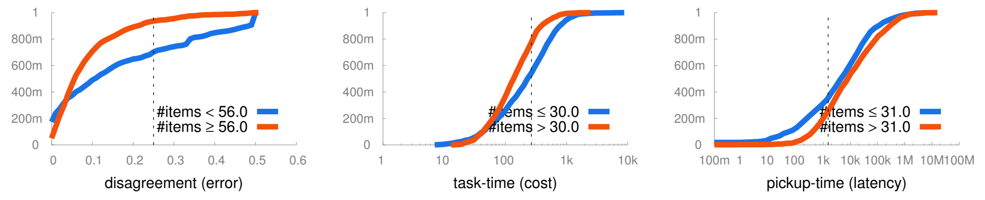
While increasing #words helps reduce disagreement in general, this benefit may be more pronounced for particular types of tasks. Intuitively, we expect detailed instructions to help more for harder tasks, and have less impact on easier tasks. To test this hypothesis, we separate tasks into buckets by their labels (recall goal, operator and data), and test the effect of our feature, #words. From Figure 8a, we see that for (relatively hard) gather tasks, #words has a pronounced effect on disagreement with higher #words leading to significantly lower disagreement. On the other hand, Figure 8b



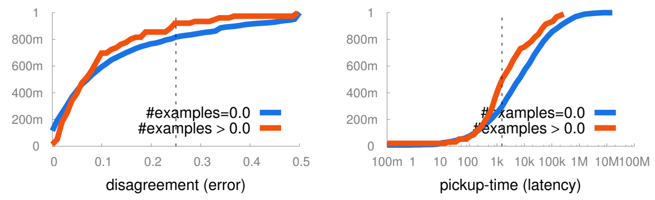
(a) CDF: #words vs disagreement



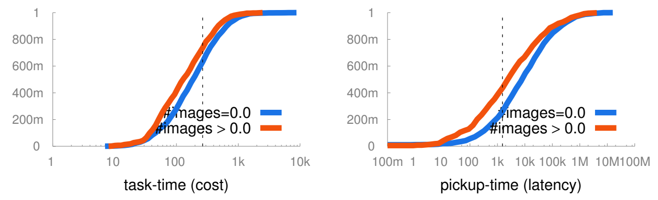
(b) CDF: #text-box vs (1) disagreement, (2) task-time



(c) CDF: #items vs (1) disagreement, (2) task-time, (3) pickup-time



(d) CDF: #examples vs (1) disagreement, (2) pickup-time



(e) CDF: #images vs (a) task-time, (b) pickup-time

Figure 7: Task Design Parameters and Metrics

seems to indicate that for (relatively simple) rating tasks, #words has no significant impact on disagreement.

4.4 Presence of input text-boxes

Next, we explore the effect of including text boxes as input fields. We denote the number of text boxes present in the HTML interface as #text-box, and show its effect on disagreement in Figure 7b. Specifically, we compare the set of tasks having non-zero text-boxes, i.e. #text-box > 0, against tasks with no text-boxes, i.e. #text-box $= 0$. Not surprisingly, Figure 7b shows that there tends to be higher disagreement between workers for tasks with text-boxes. This could be due to the fact that disagreement is agnostic to the input operator type and looks for an exact match of worker answers, while also possibly being affected by the fact that textual tasks may be more subjective (we have however, filtered out all tasks with very high disagreement). We also observe that workers tend to take longer to complete such tasks. Again, this is not surprising, as we expect it to typically take longer to fill out text than to choose from a list of options.

As in Section 4.3, we match tasks based on their labels and dig deeper to check if the insights obtained from our correlation analyses on the complete dataset hold true on individual classes of tasks as well. From Figure 8c, we see that for sentiment analysis tasks, the presence of text-boxes significantly increases the task-time. Checkboxes or multiple-choice style interfaces are likely to yield much lower task-times than ones based on text-boxes.

4.5 Number of items

Another parameter of interest is how many items are operated on in a batch across many instances and questions. Anecdotally, the number of items in a batch is known to attract workers, since they can read instructions once and work for longer without having to switch context. We use #items to denote this feature. We observe that when the #items is increased, both the task-time as well as the disagreement metrics improve. That is, tasks get done faster, and workers show lower disagreement when tasks have a higher #items (see Figure 7c). One potential reason for this is that tasks with high #items attract better and more serious workers. Another explanation is that workers get better with experience (both faster and more accurate). Increasing #items, however, has the effect of increasing the pickup-time of a task—this is probably due to the fact that even though there may be a higher #items and possibly task instances, the number of available workers (and therefore the parallelism) is fixed, and therefore the same worker may end up working on different instances in sequence, leading to higher pickup times for the task instances later on in the worker's sequence.

Further, we believe that having larger #items would help more for harder tasks, and have less impact on easier tasks. This is supported by our observations from Figure 8e. We see that #items has a pronounced effect on disagreement for (relatively hard) gather tasks with higher #items leading to significantly lower disagreement. Figure 8f on the other hand, indicates that for (simpler) rating tasks, #items has insignificant impact on disagreement.

4.6 Using examples

It is well-known that examples can have a huge influence on the effectiveness of a task, by training workers on how to answer questions. To study how many examples are used in a task, we count the number of times the word "example" comes wrapped in a tag of its own in the HTML, indicating that the example is prominently displayed — we denote this parameter by #examples. Figure 7d demonstrates that examples have the effect of improving worker agreement. We also observe that examples have the effect of reducing pickup times. It is possible that workers are more inclined to pick up ones that seem more "well-defined" or clear, thereby choosing the ones with examples preferentially over others. We observe no significant correlation between the #examples and the task-time — this may be because, the time taken to read and understand examples trades off against the improved speed of performing tasks "post-training". Finally, we match tasks based on their labels and dig deeper into individual categories of tasks. From Figure 8d, we see that examples have a significant effect on disagreement for the most popular task goal, Language Understanding.

4.7 Adding images

We speculate that images can play a role in capturing worker interest, and improving the overall worker experience. To evaluate this aspect, we first count the number of image tags present in the HTML source—we denote this feature as #images. We find that around 700 clusters contain at least one image, while around 2200 contain none. Figure 7e shows that tasks with #images > 0 are picked up faster than those with #images $= 0$. We believe that this is due to a similar reason as with #examples — workers are attracted to more interesting and well-designed tasks, and images go a long way to help with that. We also drill-down our dataset on task categories to check if the above insight holds true even for specific categories. We plot our observation for tasks with (i) operator Extract in Figure 8g, or (ii) goal Data Quality Control in Figure 8h. These categories have a significant number of tasks with and without images and the figures show that our hypothesis that tasks are picked up faster due to the presence of images holds true even when we focus on particular operators or goals.

We also observe that tasks with images tend to get completed faster. One possible explanation for this is that for tasks with #images > 0, workers are more energetic or "enthusiastic" in completing the task, and visual understanding often takes less time than textual understanding. We observe no significant correlation between the #images and the disagreement of tasks, indicating that these tasks are not inherently easier.

4.8 Summary from a metric point of view

In addition to the features we have seen so far, we also looked for correlations between other features and the target metrics. For instance, we examined whether batches were issued on weekdays or weekends, what time of day they were issued at, and how many input fields they had. We observed no significant correlations between these features and any of our metrics. (Recall that for a correlation to be considered statistically significant, we perform a t-test and only those observations with a sufficiently small p-value are considered. For the correlations that we summarize for each of our metrics, the p-values are all significantly below our threshold of 0.01.) We present the quantitative observations corresponding to the noticeable correlations in Tables 1, 2 and 3, and discuss the underlying insights below.

Disagreement Score. Table 1 summarizes the effect of features that show correlation with the disagreement of tasks. Based on our observations, we draw the following conclusions: Providing detailed instructions for workers can be crucial. If we have multiple items or questions, we should issue them together in one batch (as opposed to scattered across batches) in order to benefit from more experienced workers and workers who get better with experience. Interfaces should use multiple-choice questions to phrase tasks rather than text-based ones wherever possible. Examples are also crucial in reducing errors.

Median Task Time. Table 2 summarizes the effect of features that show correlation with the task-time of tasks. Based on our observed correlations, we note that similar to disagreement, it is beneficial to issue items all at once to benefit from workers with experience. Interfaces should use multiple-choice questions to phrase tasks rather than text-based ones wherever possible, as they also affect the typical task time, and correspondingly, worker effort. Adding images not only makes tasks look more pleasing, but also improves worker experience and latency.

Table 1: Disagreement Score: summary

Feature	Cluster Bins (split at median(feature-value))				disagreement	
	Bin-1	# clusters	Bin-2	# clusters	Bin-1	Bin-2
#words	≤ 466	1150	> 466	1149	0.147	0.108
#items	< 56	1148	≥ 56	1151	0.169	0.086
#text-boxes	$= 0$	1283	> 0	1014	0.102	0.160
#examples	$= 0$	2221	> 0	76	0.128	0.101

Table 2: Median Task Time: summary

Feature	Cluster Bins (split at median(feature-value))				task-time	
	Bin-1	# clusters	Bin-2	# clusters	Bin-1	Bin-2
#items	≤ 30	1511	> 30	1469	230s	136s
#text-boxes	$= 0$	1565	> 0	1412	119.0s	285.7s
#images	$= 0$	2268	> 0	709	183.6s	129.0s

Median Pickup Time. Table 3 summarizes the effect of features that show correlation with the pickup-time of tasks. Including examples and images is observed to help increase pick-up rate (reduce latency), probably because workers are attracted to more interesting and well-structured tasks. At the same time, issuing more task instances in parallel will lead to increases in the pickup time due to limited parallelism in the marketplace.

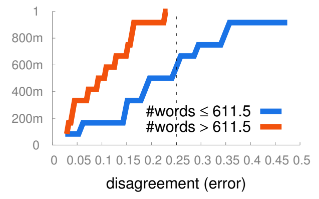
4.9 Predictive Setting

We further concretize our findings from the previous section by exploring the use of the features for prediction. We demonstrate that using just these features allows for an accurate approximate estimation of various metrics. Due to the high variability in the range of values of our metrics, it is not possible to predict the exact value of a metric for any given task. Instead, we bucketize the range of values into 10 buckets, and try to predict which bucket any given task will fall into. For example, instead of trying to predict disagreement for a given task, we predict whether the disagreement would fall into the buckets $[0, 0.1), [0.1, 0.2), \ldots, [0.9, 1.0]$. There are many different ways in which we could bucketize the range of values—each bucketization also corresponds to distributing tasks into buckets. In the following, we shall use the term bucketization to refer to the bucketization of the metric's range of values, as well as tasks interchangeably. In our experiments, we consider the two most natural ones: (1) bucketization by range, where we evenly divide the range of metric values into buckets of uniform width, and (2) bucketization by percentiles, where we divide the range of metric values into buckets such that all buckets contain roughly equal number of tasks. For each of these two cases, we divide all three of our metrics into 10 buckets. We run a simple decision tree classifier with the following feature sets: (1) features for disagreement: {#items, has-example, #words, #text-boxes}, (2) features for task-time: {#items, has-image, #text-boxes}, (3) features for pickup-time: {#items, has-example, has-image}. We perform a 5-fold cross-validation to test the accuracy of our models.

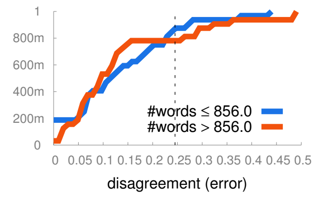
Bucketization by range. We observe that we are able to predict the *exact* bucket for tasks of disagreement with accuracy 39%, of task-time with 95%, and of pickup-time with 98%. Note that here accuracy is averaged across the 5 test cases in our cross-validation. For disagreement, we obtain an high accuracy of 62% if we allow an error tolerance of 1 bucket—that is, using just these features alone, we are able to predict within a tolerance of 1 bucket the

Table 3: Median Pickup Time: summary

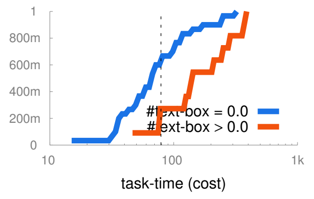
Feature	Cluster Bins (split at median(feature-value))				pickup-time	
	Bin-1	# clusters	Bin-2	# clusters	Bin-1	Bin-2
#items	≤ 31	1471	> 31	1470	4521s	8132s
#examples	$= 0$	2845	> 0	93	6303s	1353s
#images	$= 0$	2230	> 0	708	7838s	2431s



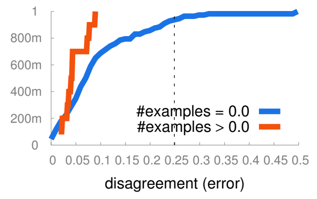
(a) #words-disagreement: Gather



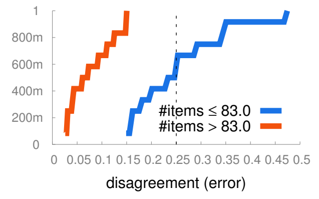
(b) #words-disagreement: Rating



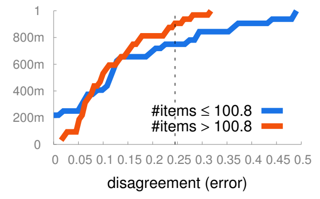
(c) #text-boxes-task-time: SA



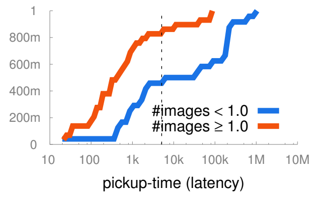
(d) #examples-disagreement: LU



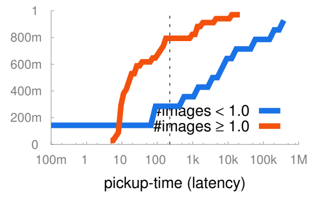
(e) # items-disagreement: Gather



(f) # items-disagreement: Rating



(g) #images-pickup-time: Extract



(h) #images-pickup-time: Quality

Figure 8: Features-Metrics CDF: Drill down by match on labels

disagreement for majority of the tasks on average. Given the extremely high dimensional nature of this prediction problem, with a very large number of hidden variables that we have not considered, even the 39% accuracy seen for disagreement is very high. To verify that the accuracies for task-time and pickup-time are not heavily biased by a skew in the distribution of tasks across buckets for these metrics, we also perform a similar cross-validation test for the percentile-based bucketization. We observe that even in this harder case, our model is able to make predictions with reasonable accuracy—we discuss the percentile-bucketization setting in our technical report [24].

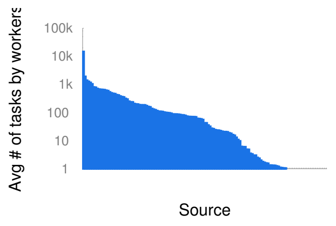
5. WORKER ANALYSES

In this section, we adopt a worker-centric view of the marketplace and evaluate the worker demographics and behavior patterns. Specifically, we look at (1) distribution of workers across different sources and regions, (2) lifetimes and attention spans of workers.

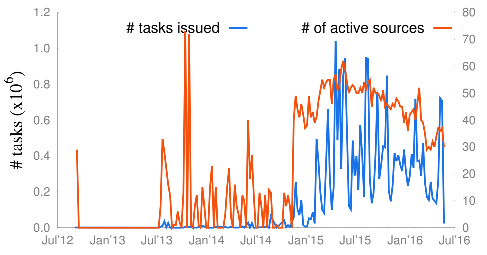
5.1 Where do the workers come from?

Labor Sources. As described earlier, the marketplace we focus on, unlike Mechanical Turk, gets crowd workers from multiple sources: specifically, the marketplace has *139 different sources* for crowd labor, altogether supplying around 69,000 workers across the period of our collected data. These sources—all distinct from each other—are listed in our technical report [24]. These sources all link to, and allow workers to sign-up with the marketplace. The marketplace directly compensates workers through one of many mechanisms: money, gift cards, or bitcoins. Some of these sources (e.g., imerit_india, yute_ jamaica, taskhunter) are specific to certain locations in the world, while others provide workers tailored to specific domains of tasks (e.g., ojooo provides workers for advertising and marketing campaigns). In addition, the marketplace also has its own dedicated worker pool (called internal), performing $484k$ tasks, that is about 2% of all tasks in our collected sample.

We also plot the average number of tasks performed by workers on different sources in Figure 9a. Each vertical splice on the x-axis represents a labor source, and the height of the splice indicates the number of tasks performed by a worker from that source on average. We see a significant variation in the worker loads across source (the y-axis is log-scale). For some sources, workers typically perform more than 10,000 tasks each whereas on the other end of the spectrum, 40% of the sources have workers performing $\leq$ 20 tasks each. The variation in number of tasks per worker suggests the presence of two types of sources — sources having (a) a dedicated workforce performing a large number of tasks per worker, and (b) an on-demand workforce, performing few tasks per worker. Thus, the availability of these two types of sources is an essential load balancing strategy — the dedicated workforce is supplemented by on-demand workforce in periods of high task load. To study this further, Figure 9b shows the number of sources active every week overlaid on the number of tasks issued. This plot seems to indicate that after January 2015, while the marketplace has a relatively fixed number of active sources, the number of tasks issued varied quite a bit. Thus, by using a combination of sources, the marketplace is able to absorb the varying task load.



(a) Average number of tasks by workers from different sources



(b) Number of sources used over the evaluation period

Figure 9: Tasks performed by workers across sources across weeks

Next, we look at the major contributing sources by (a) number of workers, and (b) number of tasks in Figure 10. In Figure 10d, we show the top 10 sources by the number of tasks performed by its workers. These 10 sources together account for $\sim$95% of the tasks, and $\sim$86% of the workers in the marketplace. Figure 10a shows the top 10 sources contributing the most number of workers. Popular sources include Clixsense [1] and NeoDev [4] — companies that provide monetary payment for users taking surveys, and

Prodege [5] — a company that rewards workers in gift cards. We note some tasks are also routed to Mechanical Turk (amt) workers, which accounts for $\sim$1.5% of all workers. While Mechanical Turk has contributed a total of $\sim$1000 workers over the period of our evaluation, with a maximum of $\sim$400 of their workers being active at any given point of time, by comparison the source NeoDev has contributed a total of $\sim$27000 workers in all with as many as $\sim$2600 of them being active in a single week. In addition, the marketplace's own internal workforce (internal) accounts for 2.5% of the total workforce and more than $484k$ tasks in our sample during the evaluation period.

Geographic distribution of workers. In a study of Amazon Mechanical Turk's workforce [23], the authors noted that more than 60% of the workers came from USA and India. For our marketplace, while these countries continue to contribute a significant number of workers, we also see 17% of workers coming from the emerging South American and African markets. Close to 50% of the workers come from 5 countries — USA (21.3k), Venezuela (5.3k), Great Britain (4.4k), India (4.1k) and Canada (2.8k). We observe that crowdsourcing has become a truly global phenomenon with workers coming from as many as 148 countries.

Quality across sources. As we noted earlier, different sources bring workers from different locations and specializing in different types of tasks. Furthermore, while some sources have a dedicated workforce performing a large number of tasks regularly, other sources supply an on-demand workforce that performs a small number of tasks occasionally. Given these variations, we investigate if the *quality* of workers varies across sources. We evaluate the quality of different sources on two metrics. Our first metric is the trust score attributed by the marketplace to each completed task performed by a single worker. We compute and report the *mean trust* assigned to tasks performed by workers from each source. The second metric measures the amount of time taken by workers to complete tasks. To normalize across different tasks, we divide a worker's time by the median time taken by workers to complete that task. We report the average of these *relative task times* for tasks performed by each source as the second metric of quality.

The variation in quality (trust and latency) for all sources is shown in Figure 10c and Figure 10f. In terms of mean trust, we observe that close to 10% of the sources have mean trust < 0.8. The trust for some sources is even lower than 0.5. The difference in quality between the sources is more evident when we look at the mean relative task times. While most sources have mean relative task times close to 1, 5% of the sources have mean relative task times ≥ 3 — the workers from these sources take more than $3\times$ time to complete the tasks, compared to median task times. Three of these sources even have mean relative task times ≥ 10.

We further examine the quality of major sources *i.e.*, sources providing the most number of workers, in Figures 10b and 10e. With the exception of Mechanical Turk (amt), these sources have high quality — having mean trust > 0.8 and mean relative task time < 1.5. Mechanical Turk performs poorly on both metrics — the mean relative task time is more than 5 and mean trust is 0.75.

5.2 How do the worker workloads vary?

As our analysis of sources indicated, most of the tasks on the marketplace are completed by a small group of workers. To explore this issue in detail and look at the distribution of worker workloads, we plot the number of tasks performed by each worker in Figure 11a. The x-axis shows the rank of the worker, when workers are sorted in decreasing order of number of tasks completed. The y-axis value shows the number of tasks completed by the worker. From the plot, we note that majority of workers' participation is one-off and the workload is mostly shared by a small group of workers. In fact, more than 80% of the tasks are completed by just 10% of the workforce. Given their experience, it might be worthwhile for marketplaces to collect periodic feedback from these workers.

5.3 How engaged are crowd workers?

Worker Lifetimes. We investigate the workers' availabilities on the marketplace through two metrics: (1) the *lifetime* of the workers, which is the number of days between their last and first activity on the marketplace during the evaluation period, and (2) the number of *working* days (out of their lifetimes) where workers have taken up tasks.

We first show the distribution of the lifetimes of the workers through a histogram in Figure 12a. Each bar in the x-axis corresponds to a lifetime range, with bar heights denoting the the number of workers having lifetime in corresponding ranges. From these plots, we note that 79% number of workers are only available over short time frames and hence have lifetimes of less than 100 days. In fact, 52.7% of the workers have a lifetime of only 1 day in the evaluation period, indicating that a majority of workers are directed to the marketplace through their sources for one-off tasks. However, these workers are not major contributors in terms of number of tasks – they complete only 2.4% of the tasks in the marketplace.

Of the remaining workers who have logged in to the marketplace on more than one day, about one-third have been working on more than 10 days and have completed 83% of the tasks in the marketplace. Next, we focus on these *active* workers and explore their behavior in greater detail.

5.4 Active Worker Characteristics

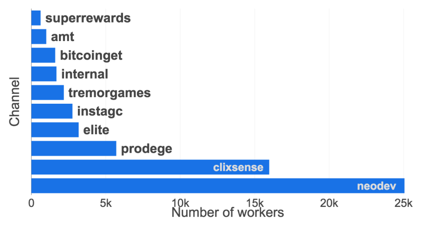
Distribution of Working Days. Consider the distribution of working days for the active workers in Figure 12b. First, on the right side of the plot, note the presence of workers who have been working on more than 350 days. This is especially remarkable, considering the fact the evaluation period contains regular data for only about 18 months. Second, the bar heights reduce close to linearly (in log scale) with the active days, indicating that the availability of workers decreases exponentially with experience.

For the active workers, we also plot a histogram of the fraction of lifetimes days where they have been working in Figure 12c. We note that among these active workers, more than 43% are working at least once a week (on average) during their lifetimes.

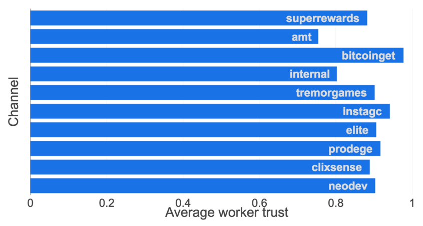
Time Spent. We use the amount of time spent by workers on tasks as a proxy for the total time spent working. While this may not be accurate because workers also have to search for tasks, this serves as a good estimate of their productive time. Figure 11b shows the total number of hours clocked by the active workers during their lifetime. We notice a skewed but long tailed distribution; $\approx$10,000 workers have been working on tasks for less than 25 hours. Nonetheless, there are also a handful of workers who have worked for more than 300 hours during the evaluation period.

Next, in Figure 11c, we plot the average number of hours spent by these active workers on a working day. From this plot, we note that more than 90% of the workers work for less than 1 hour during their working days. This suggests that crowdsourcing is still not at a scale where it can support many active workers on a full-time basis. However, more than a thousand workers still spend more than an hour a day working on tasks.

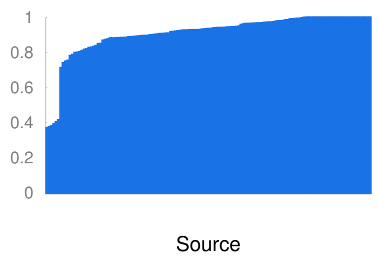
Trust. The mean and median of the average trust of active workers are both above 91%, and 90% of all active workers have average trust higher than 0.84. Given that the trust scores of workers are all



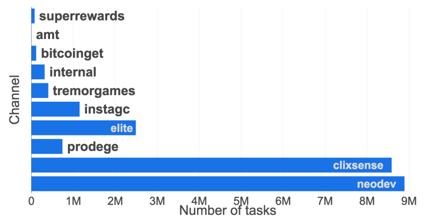
(a) Number of workers from top sources



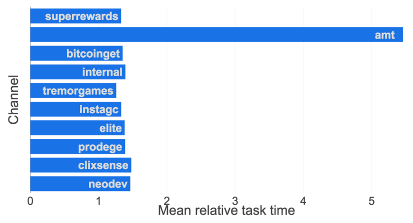
(b) Mean trust of top sources



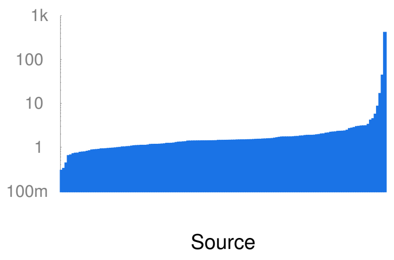
(c) Mean trust



(d) Number of tasks by top sources

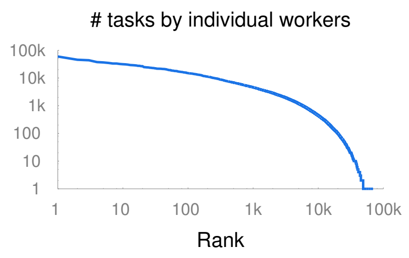


(e) Mean relative task time of top sources

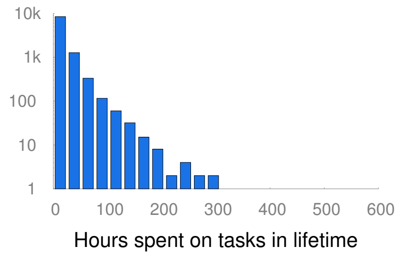


(f) Mean relative task time

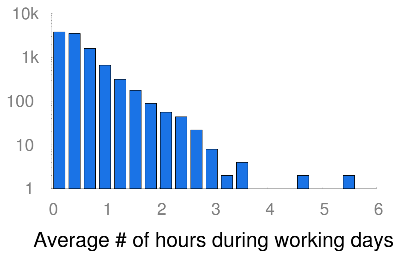
Figure 10: Number of tasks, latency and trust distributions of sources



(a) Workload distribution

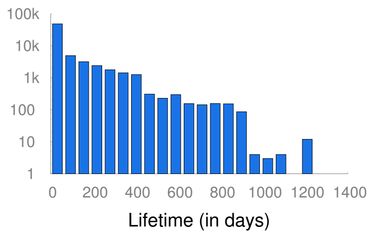


(b) Total hours spent on tasks in lifetime

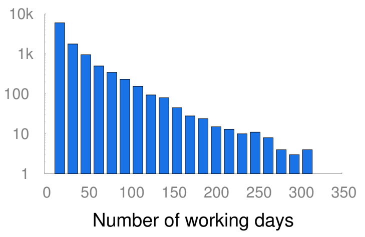


(c) Average number of hours spent on an active day

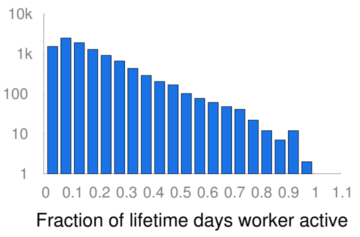
Figure 11: Workload and Time spent distributions



(a) Distribution of lifetimes



(b) Number of working days for active workers



(c) Fraction of lifetime for which workers are active

Figure 12: Worker lifetimes

so high, and show such little variation, their distribution does not yield any novel insights.

6. RELATED WORK

One of the first papers in the crowdsourcing literature focused on analyzing the Mechanical Turk marketplace for demographic factors [22, 23]; a more recent paper studied similar aspects by issuing surveys to Mechanical Turk workers [35]. Other recent papers study the motivations of crowd workers by conducting broad surveys [10, 11]. Other papers have evaluated various aspects of marketplaces by interviewing or issuing tasks to workers, such as truthfulness [39] and consistency [38], the efficacy of conducting interface evaluations via crowdsourcing [31, 25], limitations in using Mechanical Turk for experimentation [37], and challenges faced by workers with disabilities [41]. Others attempt to understand worker motivations and behavior using (Turker Nation) forum data [30], and workers' on- and off-network interactions [18]. A recent book [29] described the results of interviewing marketplace companies (including Samasource [6], Crowdflower [2]) for their concerns and problems, but did not conduct a similar quantitative study based on marketplace data. Marketplace companies sometimes publish their own reports on demographics, e.g., [40, 7, 20]. In [26], the authors discuss the challenges faced by crowdsourcing marketplaces, and describe their vision for the future.

The paper that is closest to us in adopting a data-driven approach is the one by Difallah et al. [14], focused on studying the marketplace dynamics of Mechanical Turk, such as marketplace demand and supply, evolution of task payments over time, as well as other topics. We have some overlap with the Difallah et al. paper in terms of marketplace analysis, but our emphasis and granularity is different; at the same time, the findings in our task design and worker analysis sections are entirely new [24]. Since their dataset does not have information about individual worker responses, they are unable to study the question of task "effectiveness" like we do. Also unlike that paper that focuses on Mechanical Turk, our crowdsourcing marketplace recruits workers from a collection of labor sources, making it a crowdsourcing "intermediary" or "aggregator", and allows for a number of interesting additional analyses.

Our work in gaining a better understanding of crowd work has broad ramifications for the database community who has been developing crowd-powered data processing algorithms [19, 32, 13, 36, 8, 12], and systems, e.g., [28, 17, 27, 9, 33], with dozens of papers published in database conferences each year. ([29] surveys this literature.) As examples, understanding the relative importance of various types of processing needs, can prioritize the attention of our community to unexplored or underoptimized areas; understanding how tasks are picked up and worked on can help develop better models of task latency; understanding the worker perspective and engagement can aid in the design of better models for worker accuracy and worker behavior in general; and understanding the impact of task design can help the community adopt "best practices" to further optimize cost, accuracy, and latency.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we quantitatively address several important open-ended questions aimed towards understanding and improving the paradigm of crowdsourcing from three key perspectives — marketplace dynamics (important to marketplace administrators), task design (important to requesters), and worker characterization (important to labor sources, marketplace administrators, and requesters). We answer several of what we believe are the most important open questions about crowdsourcing interactions, through quantitative, data-driven experiments. Based on our experiments, we come up with a number of valuable insights, that we hope will inform and guide the evolution of crowdsourcing over the coming years.

There are a number of directions that one could explore, following this work. A natural direction for the database community is to explore the unexplored combinations of popular task types that have not yet been adequately optimized. More broadly, it would be useful to pursue a deeper understanding of worker behavior by looking at phenomena such as worker anchoring, worker learning, and interactions between various jobs. While we have restricted our analyses to a specific set of features and metrics—a full analysis of the interplay between various different task parameters and notions of job success would be a natural next step. Lastly, with full-fledged A/B testing, we may be able to solidify our correlation and predictive claims with further causation-based evidence.

Acknowledgements: We acknowledge support from grant IIS-1513407, IIS-1633755, and IIS-1652750 awarded by the NSF, grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge initiative (www.bd2k.nih.gov), and funds from Adobe, Google, and the Siebel Energy Institute. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

8. REFERENCES

[1] ClixSense. *http://www.clixsense.com/.*

[2] CrowdFlower. *http://crowdflower.com.*

[3] Mechanical Turk. *http://www.mturk.com.*

[4] NeoDev. *http://www.neodev.se/.*

[5] Prodege. *http://www.prodege.com/.*

[6] Samasource. *http://samasource.com.*

[7] Samasource Jobs. *http://www.samasource.org/people/#jobs.*

[8] Y. Amsterdamer et al. Crowd mining. In *SIGMOD*, 2013.

[9] A. Bozzon, M. Brambilla, and S. Ceri. Answering search queries with crowdsearcher. In *WWW*, pages 1009–1018, 2012.

[10] A. M. Brawley and C. L. Pury. Work experiences on mturk: Job satisfaction, turnover, and information sharing. *CHB*, 2016.

[11] R. Brewer et al. Why would anybody do this?: Understanding older adults' motivations and challenges in crowd work. In *CHI*, 2016.

[12] A. Das Sarma et al. Towards globally optimal crowdsourcing quality management: The uniform worker setting. In *SIGMOD*, 2016.

[13] S. B. Davidson, S. Khanna, T. Milo, and S. Roy. Using the crowd for top-k and group-by queries. In *ICDT*, pages 225–236, 2013.

[14] D. E. Difallah et al. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In *WWW*, pages 238–247, 2015.

[15] *http://www.behind-the-enemy-lines.com/2016/02 /a-cohort-analysis-of-mechanical-turk.html*. Cohort analysis of mturk requesters, blog post, 2015.

[16] *http://www.forbes.com/sites/elainepofeldt/2015/05/05 /elance-odesk-becomes-upwork-today-odesk-brand-gets-phased-out*. Odesk brand gets phased out, forbes.com, 2015.

[17] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin. Crowddb: answering queries with crowdsourcing. In *SIGMOD*, 2011.

[18] M. L. Gray, S. Suri, S. S. Ali, and D. Kulkarni. The crowd is a collaborative network. In *CSCW*, pages 134–147. ACM, 2016.

[19] S. Guo et al. So who won?: dynamic max discovery with the crowd. In *SIGMOD Conference*, pages 385–396, 2012.

[20] https://www.odesk.com/oconomy/activity/. Odesk oconomy, 2012.

[21] https://www.upwork.com/blog/2013/12/mergerfaq/. Odesk-elance merger faq, upwork blog, 2014.

[22] P. G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS*, 17:16–21, December 2010.

[23] P. G. Ipeirotis. Demographics of mechanical turk. 2010.

[24] A. Jain et al. Understanding workers, developing effective tasks, and enhancing marketplace dynamics. *http://data-people.cs.illinois.edu/papers/crowd-data.pdf.*

[25] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *CHI*, pages 453–456, 2008.

[26] A. Kittur et al. The future of crowd work. In *CSCW*, 2013.

[27] X. Liu et al. Cdas: A crowdsourcing data analytics system. *PVLDB*, 5(10):1040–1051, 2012.

[28] A. Marcus et al. Crowdsourced databases: Query processing with people. In *CIDR*, pages 211–214, 2011.

[29] A. Marcus and A. Parameswaran. Crowdsourced data management: Industry and academic perspectives. *Found. Trends databases*, 6(1-2).

[30] D. Martin et al. Being a turker. In *CSCW*. ACM, 2014.

[31] W. Mason and S. Suri. Conducting behavioral research on amazon's mechanical turk. *Behavior research methods*, 44(1):1–23, 2012.

[32] A. Parameswaran et al. Optimal crowd-powered rating and filtering algorithms. *VLDB*, 2014.

[33] H. Park et al. An overview of the deco system: Data model and query language; query processing and optimization. *ACM SIGMOD Record*, 41, 2012.

[34] A. Ramesh et al. Identifying reliable workers swiftly. Technical report, Stanford InfoLab, 2012.

[35] J. Ross et al. Who are the crowdworkers?: shifting demographics in mechanical turk. In *CHI Extended Abstracts*. ACM, 2010.

[36] A. D. Sarma, A. Parameswaran, H. Garcia-Molina, and A. Halevy. Crowd-powered find algorithms. In *ICDE*, 2014.

[37] N. Stewart et al. The average laboratory samples a population of 7,300 amazon mechanical turk workers. *JDM*, 2015.

[38] P. Sun and K. T. Stolee. Exploring crowd consistency in a mechanical turk survey. In *CSI-SE*. ACM, 2016.

[39] S. Suri et al. Honesty in an online labor market. In *Human Computation*, 2011.

[40] N. Zukoff. Demographics of the Largest On-demand Workforce. http://www.crowdflower.com/blog/2014/01 /demographics-of-the-largest-on-demand-workforce, 2014.

[41] K. Zyskowski et al. Accessible crowdwork?: Understanding the value in and challenge of microtask employment for people with disabilities. In *CSCW*, pages 1682–1693. ACM, 2015.