Making Better Use of the Crowd*

Jennifer Wortman Vaughan
Microsoft Research, New York City
jenn@microsoft.com

December 5, 2016

1 Introduction

Over the last decade, crowdsourcing has been used to harness the power of human computation to solve tasks that are notoriously difficult to solve with computers alone, such as determining whether or not an image contains a tree, rating the relevance of a website, or verifying the phone number of a business.

The machine learning community was early to embrace crowdsourcing as a tool for quickly and inexpensively obtaining the vast quantities of labeled data needed to train machine learning systems. For example, in their highly influential paper, Snow et al. [59] used crowdworkers to annotate linguistic data for common natural language processing tasks such as word sense disambiguation and affect recognition. Similar ideas were applied to problems like annotating medical images [53] and discovering and labeling image attributes or features [51, 52, 73]. This simple idea—that crowds could be used to generate training data for machine learning algorithms—inspired a flurry of algorithmic work on how to best elicit and aggregate potentially noisy labels [15, 23, 28–30, 42, 58, 67, 71, 72], and is probably what many people in the machine learning community think of when they think of crowdsourcing.

In the majority of this work, it is assumed that once collected, the labeled data is handed off to a machine learning algorithm for use in training a model. This handoff is typically where the interaction with the crowd ends. The idea is that the learned model should be able to make autonomous predictions or actions. In other words, the crowd provides the data, but the ultimate goal is to eventually take humans out of the loop.

This might lead one to ask: *What other problems could the crowd solve?*

In the first half of this tutorial, I will showcase innovative uses of crowdsourcing that go far beyond the collection of labeled data. These fall into three basic categories:

- **Direct applications to machine learning.** For example, the crowd can be used to evaluate machine learning models [9], cluster data [18, 62], and debug the large and complex machine learning models used in fields like computer vision and speech recognition [46, 47, 50].

*These notes—part survey, part position paper, part best practice guide—were written to accompany the NIPS 2016 tutorial *Crowdsourcing: Beyond Label Generation* and follow the same general outline.

- **Hybrid intelligence systems.** These "human in the loop" AI systems leverage the complementary strengths of humans and machines in order to achieve more than either could achieve alone. While the study of hybrid intelligence systems is relatively new, there are already compelling examples that suggest their great potential for applications like real-time on-demand closed captioning of day-to-day conversations [35–38, 48], "communitysourced" conference planning [3, 12, 32], and crowd-powered writing and editing [5, 31, 33, 56, 64].
- **Large scale studies of human behavior online.** Crowdsourcing is gaining popularity among social scientists who use platforms like Amazon Mechanical Turk to quickly and easily recruit large pools of subjects for survey-based research and behavioral experiments. Such experiments can benefit computer science too. With the rise of social computing, computer scientists can no longer ignore the effects of human behavior when reasoning about the performance of computer systems. Experiments allow us to better model things like how humans perceive security threats Ur et al. [65], understand numbers [4], and react to annoying advertisements [16], which leads to better designed algorithms and systems.

Viewed through another lens, we can think of these three categories of applications as illustrating the potential of crowdsourcing to influence machine learning, AI systems more broadly, and finally, all of computer science (and even fields beyond computer science).

In the second half of the tutorial, I will talk about one of the most obvious and important yet often overlooked aspects of crowdsourcing: *The crowd is made of people.*

I will dive into recent research aimed at understanding who crowdworkers are, how they behave, and what this should teach us about best practices for interacting with the crowd.

I'll start by debunking the common myth among machine learning researchers that crowdsourcing platforms are riddled with bad actors out to scam requesters. In particular, I'll describe the results of a research study that showed that crowdworkers on the whole are basically honest [61].

I'll talk about experiments that have explored how to boost the quality and quantity of crowdwork by appealing to both well-designed monetary incentives (such as performance-based payments [22, 24, 68, 69]) and intrinsic sources of motivation (such as piqued curiosity [39] or a sense of meaning [8, 54]).

I'll then discuss recent research—both qualitative [19] and quantitative [70]—that has opened up the black box of crowdsourcing to uncover that crowdworkers are not independent contractors, but rather a network with a rich communication structure.

Taken as a whole, this research has a lot to teach us about how to most effectively interact with the crowd. Throughout this part of the tutorial I'll discuss best practices for engaging with crowdworkers that are rarely mentioned in the literature but make a huge difference in whether or not your research studies will succeed. (Here's a few hints: Be respectful. Be responsive. Be clear.)

Crowdsourcing has the potential for major impact on the way we design, test, and evaluate machine learning and AI systems, but to unleash this potential we need more creative minds exploring novel ways to use it. This tutorial is intended to inspire you to find novel ways of using crowdsourcing in your own research and to provide you with the resources you need to avoid common pitfalls when you do.

2 The Potential of Crowdsourcing

In the first half of this tutorial, I will walk through a wide range of innovative applications of crowdsourcing that go beyond the collection of data. I'm using the term "crowdsourcing" very generally here to encompass both paid and volunteer crowdwork, done by experts or nonexperts, on any general or specialized crowdsourcing platform. At this point, I want to avoid committing to any specific definition of what crowdsourcing is and suggest that you interpret it in the broadest sense.

2.1 Direct Applications to Machine Learning

Since most people here are machine learning researchers, let me start by describing a few direct applications of crowdsourcing to machine learning.

2.1.1 Crowdsourcing Labels and Features

While I won't dwell on it, I'd be remiss to avoid mentioning the application that first got the machine learning community excited about crowdsourcing: generation of labeled data. The way that this usually works is that crowdworkers are presented with unlabeled data instances (such as websites or images) and are asked to supply labels (for instance, a binary label indicating whether or not the website contains profanity or a list of keywords describing the content of the image). Since the supplied labels can be noisy, the same instances may be presented to multiple crowdworkers and the workers' responses combined [15, 23, 28–30, 42, 58, 67, 71, 72]. This approach has been applied to collect data for natural language processing [59], computer vision [53], and many other fields. One of the behavioral studies I'll discuss later uses crowdsourced data labeling as a first step in a more complex experiment.

Crowdsourcing can also be used to identify and subsequently label diverse sets of salient features [73] such as attributes of images [51, 52]. The advantage of using a crowd over automated techniques is the ability to discover features that rely on knowledge and background experience unique to humans. For example, if a data set consists of images of celebrities, a human might use their background knowledge to define features such as "actor," "politician," or "married."

2.1.2 Crowd Evaluation of Learned Models

One application of crowdsourcing that has really taken off in some segments of the machine learning community is the use of crowdsourcing to evaluate learned models. This is especially useful for unsupervised models for which there is no objective notion of ground truth.

As an example, let's think about topic models. A topic model discovers thematic topics from a set of documents, for instance, New York Times articles from the past year. In this context, a *topic* is a distribution over words in a vocabulary. Every word in the vocabulary occurs in every topic, but with different probability or weight. For example, a topic model might learn a food topic that places high weight on `cheese`, `kale`, and `bread`, or a politics topic that places high weight on `election`, `senate`, and `bill`.

Topic models are often used for data exploration and summarization. In order to be useful in these contexts, the learned model should be human-interpretable in the sense that the topics it discovers should make sense to people. However, human interpretability is hard to quantify, and as a result, topic models are often evaluated based on other criteria such as predictive power.

Chang et al. [9] proposed using crowdsourcing as a way to measure the interpretability of a set of topics. In particular, they designed a *word intrusion* task in which a crowdworker is presented with a randomly ordered list of the most common words from a topic along with one intruder word that has low weight for that topic but high weight for another topic. The worker is then asked to identify the intruder. If the topic is coherent, then picking out the intruder should be easy (think {`cheese`, `bread`, `steak`, `election`, `mushroom`, `kale`}). If not, the intruder would be harder to pick out. The average error that crowdworkers make on this task can thus be used as a proxy for how interpretable or coherent topics are. Chang et al. found that previous measures of success like high log likelihood of held out data do not necessarily imply human interpretability.

As the idea of using crowdsourcing to evaluate topic models caught up, researchers also began thinking about how to incorporate feedback from crowds to improve the model over time. For example, Hu et al. [26] allow crowdworkers to suggest constraints capturing words that should be associated with the same topic but are not, and then use these constraints to improve the model.

Crowdsourcing has also been used, for instance, for evaluation of translations [7] and for relevance evaluation in information retrieval tasks [2].

2.1.3 Human Debugging of Machine Learning Models

In fields like computer vision, speech recognition, translation, and natural language processing, systems often consist of several discrete components linked together to perform a complex task. For example, consider the problem of semantic segmentation which involves partitioning an image into multiple semantically meaningful parts and labeling each part with a class. There are promising approaches to this problem that use conditional random fields (CRFs) or other machine learning models to integrate feedback from independent components that perform various scene understanding tasks like object detection, scene recognition, and segmentation.

If a system designer wants to improve performance, it is not always clear which component to focus attention on. To solve this problem, Parikh and Zitnick [50] proposed the clever idea of *human debugging*, in which humans are used to uncover bottlenecks in AI systems, and applied this idea to several problems in computer vision.[1]

Human debugging helps identify which component in a system is the "weakest link." The basic idea is simple. To quantify how much an improvement to a particular component would benefit the system as a whole, we could imagine replacing this component with something (close to) perfectly accurate and testing how much the system improves. Since for many vision and language tasks human performance is an upper bound on what we might expect from a machine, we could replace the component with a human instead.

Mottaghi et al. [46, 47] applied this idea in order to analyze the performance of a CRF that has been used in the computer vision community for scene understanding. They replaced each component with crowd-

[1] Kovashka et al. [34] have a recent survey of crowdsourcing applied to computer vision tasks.

workers from the popular crowdsourcing platform Amazon Mechanical Turk and measured the change in performance of both the component in isolation and the system as a whole.

One of their most interesting findings was that humans are actually less accurate than machines at one particular task (classifying super-pixels) yet when human classifications were plugged into the CRF, the system performance improved. This suggests that making fewer mistakes classifying super-pixels is not enough. Rather it is more important that the classifier makes the right kind of mistakes. This kind of feedback helps designers know where to focus their effort.

2.1.4 Crowdsourcing Similarity Functions and Crowd Clustering

I'll mention a couple of examples of recent work that showed how crowdsourcing can be applied to solve unsupervised learning tasks like estimating some notion of object similarity or clustering objects.

Similarity matrices, which assign similarity scores to pairs of objects, are useful for exploratory data analysis, data visualization, clustering, or classification using kernel-based approaches such as support vector machines. There are automated techniques for discovering similarities, but these can fail to uncover similarities that rely on specific semantic knowledge or experience that is unique to humans. Returning to the previous example of a celebrity image data set, a human might consider background knowledge about a celebrity's profession or home country when determining how similar two celebrities are.

Tamuz et al. [62] considered the problem of estimating a similarity matrix over all pairs of n objects from human judgments. They proposed an adaptive algorithm based on comparisons of triples which is able to learn a similarity matrix with a relatively small number of human judgments. Using their approach, they were able to answer questions like which necktie would be a good substitute for another, a task that would perhaps be difficult for a machine without specialized human knowledge.

Around the same time, Gomes et al. [18] suggested an approach to the problem of crowd-based clustering that involves presenting relatively small sets of objects to each member of the crowd and asking for a clustering of these objects. These partial clusterings are then used to train a Bayesian model with the goal of producing a full clustering over all objects.

For these applications, providing good instructions and clear guidance to workers is key. If the discovered features or clusters are intended to be used towards a certain goal, communicating this goal to the crowd can help them to identify the most salient aspects of the data on which to focus.

We will return to the idea of crowdsourced clustering later when we discuss hybrid intelligence systems.

2.2 Hybrid Intelligence Systems

A *hybrid intelligence system*[2] is a "human in the loop" AI system made up of both human components and machine components. These systems are designed to leverage the complementary strengths of humans and machines with the hope of accomplishing more than would be possible using humans or machines alone. I'll next describe a couple of particularly compelling hybrid intelligence systems.

[2]Other common terms include human-in-the-loop systems and mixed initiative systems. I believe I started using the phrase hybrid intelligence after hearing it used by Ece Kamar [27].

2.2.1 Hybrid Intelligence for Speech Recognition

Let me start with an example of a system that I really like that was built by Walter Lasecki, Jeff Bigham, and colleagues [35–38, 48]. This system addresses the problem of closed captioning. Closed captioning is something that can be done reasonably well using existing speech recognition techniques under ideal circumstances, for example, when the voice recording is high quality and the system has been trained on data from the particular speaker. It does not work as well on low quality audio, with novel speakers, with speakers with heavy accents, or with language that contains a lot of technical jargon. For these scenarios, the best results come from hiring a professional stenographer, but high quality stenographers can charge as much as $200-$300 an hour and are not available on demand.

The question that these researchers asked is whether it would be possible to provide less expensive, real-time, on-demand closed captioning to users who need help understanding lectures, meetings, or other day-to-day conversations.

To achieve this, Lasecki et al. built a hybrid intelligence system that makes use of cutting edge techniques from natural language processing and crowdsourcing. Here is the basic idea. When a user would like to obtain closed captions, he starts recording. The audio is sent simultaneously to several crowdworkers. These workers aren't expected to be able to fully transcribe the speech, which is way too fast to transcribe on a normal keyboard. Instead, each worker transcribes sentence fragments. The system adjusts the speed and volume of the speech to focus each worker's attention on distinct overlapping components. The system then uses language models and AI techniques to combine the workers' text into one complete and coherent stream that is delivered back to the user with a delay of only a couple seconds.

If the crowdworkers are domain experts—for example, work study students generating closed captions for a technical math or science class—the system would be capable of capturing all of the jargon that is hard to get right using traditional automated closed captioning systems.

I like this example because it's a system that is really able to take advantage of the complementary strengths of people and of machines to achieve high quality results.

It is also an example of a broader phenomenon of using crowdsourcing to compensate for poor AI in the short term. Maybe one day the AI will be good enough for personalized closed captioning with low quality audio, but using the crowd allows us to achieve this faster.

2.2.2 Hybrid Intelligence for Constrained Optimization

The next hybrid system I want to discuss is Cobi[3] [3, 12, 32], a system for conference planning. Cobi is based on the idea of *communitysourcing*: it draws on the specialized expertise of people within a research community to plan out conference schedules.

The team that developed Cobi is part of the CHI community. CHI is a huge conference. In 2013, the year that Cobi was deployed, it accepted around 400 papers which were to be presented in 16 parallel tracks. Scheduling talks for this many papers is a huge feat. The organizers would like to minimize conflicts between sessions so that participants can see all of the talks they're interested in. Essentially they want to

[3] http://projectcobi.com

solve one big constrained optimization problem, but without direct access to the constraints—in this case, the overlap of interest in different papers and attendees' preferences more generally.

Cobi was designed to efficiently collect this information from the community. It includes individual components that elicit preferences, hard and soft constraints, and similarity judgments. Using this information, it provides an interface backed by optimization tools that chairs can use to fine tune the schedule to satisfy as many constraints as possible and produce something coherent.

As one example of how the crowd is used, authors are presented with lists of papers that are potentially related to theirs and asked for judgments about which papers belong in the same session and which papers should not be scheduled at the same time. Authors are motivated to provide this information because it is in their own best interest for their talk to land in a coherent session without any major timing conflicts. When Cobi was used at CHI, the authors of 87% of accepted submissions opted to provide feedback.

Where do the lists of potentially related papers come from? This is where the crowdsourced similarity judgments come in. In fact, the designers of Cobi treat this as a clustering problem and solve it using crowd clustering techniques like the ones we discussed earlier.

I want to stress that the conference chairs always maintain control, so in addition to the constraints coming from the community, they are able to use their own specialized knowledge to reason about tradeoffs between constraints and make sure the resulting schedule is sensible on the whole.

2.2.3 Hybrid Intelligence for Writing

Let me move on to another application: writing [56, 64]. Over the last few years, multiple hybrid intelligence systems have been introduced that aim to utilize crowd workers to speed up and improve the quality of writing. These include Soylent [5], Crowdforge [33], and Mechanical Novel [31], among others.

When I first heard about this work, I was skeptical. I can be a bit of a control freak about writing, as any of my coauthors will tell you. Would I really trust a crowd to be involved in the writing process? Would the final product be coherent? And doesn't writing require the unique knowledge and insight of the author?

Before I get into the details of how a hybrid intelligence system for writing might work, let's step back and take a moment to think about the process of writing. I recently saw a talk by Jaime Teevan from Microsoft Research that changed the way that I think about it. Teevan is an advocate of what she calls "selfsourcing" [63]. Selfsourcing takes one of the primary principles of crowdsourcing—breaking a large task into many bitesize microtasks, each of which can be completed in isolation—and applies it to our every-day attempts to get work done.

Teevan and her collaborators argue [64] that writing is most effectively done as a three-step process:

1. **Collect content.** Often the most difficult part of writing is getting started. All of us have had the experience of staring at a blank screen trying to figure out where to begin. The writing process is much easier if we start off with a more manageable and less intimidating task, such as generating short bursts of content. This content can be on the level of individual ideas that we'd like to communicate, and we can generate these ideas over time whenever they come to us.
2. **Organize content.** Once these ideas have been generated and we're no longer staring at a blank

screen, we can try to organize this content. This can involve first clustering the content by theme and then sorting the clusters in a logical order.

3. **Turn content into writing.** Next, we can begin the actual writing by turning each cluster of ideas into a coherent paragraph or section. Once this has been done, we find ourselves with a full draft, which we can edit, polish, and finalize as usual.

So far I've been talking about a single author completing each of these steps in isolation, but this doesn't have to be the case. The same process could easily be completed by a set of collaborators who independently generate initial ideas, organize content, and turn clusters of content into writing.

Taking it one step further, some of these steps—organizing content, turning content into initial paragraphs of text, perhaps some of the editing that follows—do not need to be completed by someone who is an expert on the topic. It is easy to imagine that these could be completed by crowdworkers, traditional machine learning or AI approaches, or some combination of the two. It's not important that these steps are done perfectly, because the author will still be involved in the final editing pass. This is the type of hybrid intelligence system that Teevan and her collaborators envision.

2.2.4 Hybrid Intelligence for Information Aggregation

The next example, combinatorial prediction markets [1, 21], is one that I've been working on actively myself for many years.

A prediction market is a financial market in which traders can buy or sell securities with payoffs that are linked to future events. For example, in an election market, traders might buy or sell a security that is worth \$1 if a particular candidate wins and nothing otherwise. If you believe that the probability of this candidate winning is x and you want to maximize your expected payoff, then you should be willing to buy this security at any price less than \$x, since with probability x you get \$1. Similarly, you should be willing to sell at any price greater than \$x. For this reason, we can think of the current market price of this security as capturing traders' collective beliefs about how likely it is that the candidate will win the election.

How is a prediction market an example of hybrid intelligence? Well, consider a prediction market for a United States Presidential election. What if we wanted to allow traders to express more complicated information than just the probability that a particular candidate will win the general election? To achieve the best possible aggregation of information, we might like to allow traders to express more specialized information such as the probability that a Democrat wins in North Carolina or the probability that a Republican wins in at least one of Ohio and Pennsylvania.

There are several challenges that arise. First, if we allow traders to construct arbitrary securities, there will be a liquidity problem. If a trader has highly specialized information, it will be difficult for her to find a counterparty to trade with. This problem can be solved by implementing the market using an automated market maker, a centralized algorithmic agent that is always willing to buy and sell any security at some price which typically depends on the history of trade [21]. However, with large and complex security space, it is computationally hard for a market maker to maintain coherent prices and keep its loss bounded [10]. Using techniques from convex optimization, we can design market makers that generate coherent prices (and therefore coherent predictions) over large outcome spaces while maintaining bounded monetary loss

for the market maker [1]. This gives us an algorithmic way to aggregate even the most specialized beliefs and information of traders.

2.2.5 Hybrid Intelligence Systems in Industry

I've chosen to focus on hybrid intelligence systems that have come out of the research community, but it's worth mentioning that human-in-the-loop systems are widely used in industry as well. To name just a few examples, Stitch Fix, which provides personalized style recommendations, trains machine learning algorithms to suggest items that a user might like and then sends the output of these algorithms to a human expert to curate and pare down.[4] Twitter employs contract workers to interpret search terms that suddenly spike in meaning, often due to recent mentions in the media or pop culture that an algorithm trained on stale data would miss.[5] PatternEx uses machine learning to identify suspicious activity that could indicate a security threat. This suspicious activity is then examined by a human, who determines whether there is a real attack, and the human's feedback is used to improve the system.[6]

2.3 Large Scale Studies of Human Behavior Online

Let's zoom out one more time and consider how crowdsourcing can benefit the larger computer science community.

Over the last few years, social scientists have increasingly turned to crowdsourcing platforms to recruit subjects to complete surveys and participate in behavioral experiments that traditionally would have been run in labs on campus. Crowdsourcing provides easy access to large and diverse pools of subjects and studies have shown that classic results from psychology and behavioral economics can be replicated by these crowdworkers [25, 49]. Crowdsourcing also allows faster iteration between the development of new theories and experimentation, allowing researchers to speed up their overall research process [44].

And while I won't focus on it so much for this tutorial, crowdsourcing allows researchers to push the boundaries of experiment design by recruiting larger pools of subjects and interacting with subjects over longer time horizons than would be possible in a traditional lab setting. (See, for example, the recent work of Andrew Mao and colleagues who examined what happened when 94 subjects each played 400 ten-round games of Prisoners Dilemma over the course of a month [43], an order of magnitude more than prior work.)

At the same time, computer scientists have begun to take more interest in conducting behavioral experiments and survey research of their own. With the increasing prevalence of social computing and other systems that depend on large scale human participation, it is not a good idea for computer scientists to ignore human behavior. By running experiments to develop better models of human behavior, we are able to design better algorithms and better social computing systems.

In this section, I'll discuss a few examples of ways in which crowdsourcing has been used to conduct different types of user studies and human subject experiments online.

[4] http://multithreaded.stitchfix.com/blog/2016/03/29/hcomp1/
[5] http://nyti.ms/2gzOo4K
[6] http://tek.io/1Vy01KB

2.3.1 Human Behavior and Security

I'll start with a simple example of a survey-based study.

It is widely known that Internet users have a tendency to choose overly predictable passwords. However, there has been relatively little research aimed at understanding how well users understand the security or predictability of the passwords they choose.

Ur et al. [65] used Mechanical Turk to conduct a study on user perception of password security. They surveyed crowdworkers to find out, for example, how these workers perceived the security of different password generation strategies and what each worker's mental model of an attacker is.

As an example, in one component of their study, they showed each worker pairs of passwords and asked the worker which was more secure. To evaluate these assessments, they compared them with a gold standard obtained by looking at the number of guesses that it would take for a modern password-cracking program to guess each password and calling one password more secure than another if the number of guesses required to crack it is more than an order of magnitude larger. They observed that many of their participants overestimated the benefit of replacing letters with numbers or symbols, incorrectly rating `p@ssw0rd` as more secure than `pAsswOrd`. They also underestimated the risk of including common keyboard patterns (e.g., `qwertyuiop`).

2.3.2 Human Behavior and the Communication of Numbers

The news is filled with numbers that are notoriously difficult to understand. Is a one hundred billion dollar cut to the United States federal budget large or small? It certainly sounds like a lot of money, but how does it compare with overall federal spending? To reason about the impact of such a budget cut, it may help to know that one hundred billion dollars is roughly 3% of the 2015 federal budget, or about a sixth of the amount that the United States spends annually on the military. It's also about 30% of the net worth of Beyoncé or about $5 for every person in New York State.

Barrio, Goldstein, and Hofman [4] used crowdsourcing to test whether people's understanding of numbers could be improved through the use of such perspectives.

The first step was to generate useful perspectives. To do this, they extracted 64 quotes containing measurements from recent New York Times front page articles. They asked crowdworkers on Amazon Mechanical Turk to generate perspectives for these quotes using a specialized template. To determine which perspectives were the most useful, they had other crowdworkers rate their usefulness. Finally they extracted the most useful perspectives according to these ratings. Here are a few examples:

> *The Ohio National Guard brought 33,000 gallons of drinking water to the region, while volunteers handed out bottled water at distribution centers set up at local high schools.*
>
> *To put this into perspective, 33,000 gallons of water is about equal to the amount of water it takes to fill 2 average swimming pools.*

> *They also recommended safety programs for the nations gun owners; Americans own almost 300 million firearms.*

To put this into perspective, 300 million firearms is about 1 firearm for every person in the United States.

Barrio and his collaborators used these top rated perspectives in a sequence of experiments designed to test whether perspectives increased three different proxies of numerical comprehension: recall, estimation, and error detection. For example, in the recall experiments, crowdworkers were randomly assigned to view news quotes either with or without an added perspective. After viewing the quotes, they were distracted with a quick game of Tetris, and then asked to recall the numbers from the quotes that they had viewed.

They authors found support for the benefits of perspectives across all experiments. As one example, 55% of participants who viewed the corresponding perspective were able to recall the number of firearms in the United States, while only 40% of those who viewed the quote without the perspective were able to do so.

This project is a user study, but it also hints at the possibility of building a hybrid intelligence system that could automatically extract numerical values from news articles and use the crowd to generate the most salient perspectives to display with each one in order to strengthen readers' comprehension.

2.3.3 Human Behavior and Online Advertising

The last example that I want to mention is some particularly creative work by Dan Goldstein and colleagues [16, 17] who used crowdsourcing to try to answer a high-impact business question. Publishers on the Internet display banner ads on their websites to generate profit. Some of these ads are clearly more annoying than others. What Goldstein and his colleagues asked is whether displaying these annoying ads is costing publishers money, and if so, whether it is possible to quantify how much money it is costing.

Their experiment involved two steps, both of which used crowdsourcing.

The goal of the first step was to identify some examples of good and bad ads. To do this, they presented workers on Mechanical Turk with ads and had the crowd rate how annoying each ad was. This is basically just a labeling task and is a fairly standard use of Mechanical Turk. Aggregating these ratings across crowdworkers, they produced lists of the most annoying and least annoying banner ads.

The results are essentially what you would expect. The good ads are relatively clean and unoffensive, ads that you could perhaps ignore. The bad ads are a bit more... in your face.

In the second step, they performed an experiment aimed at estimating how much (monetary) value or utility web users get from *not* being exposed to these annoying ads. They gave crowdworkers what looks like a normal crowdsourcing task, labeling email from the Enron email database as either spam or not spam. Next to each email, they showed either no ad, an ad that was determined to be good in step 1, or an ad that was determined to be annoying. They also varied how much they paid users for labeling each email. Workers could label as many emails as they wanted and would see a new ad each time, but always an ad of the same type. By looking at the number of emails that workers chose to classify in each treatment, Goldstein and his colleagues were able to estimate how much more money it would be necessary to pay workers to get them to perform the same number of classification tasks when exposed to bad ads as opposed to good ads or no ads at all. From that, they were able to estimate how much publishers lose by showing bad ads.

I won't go into detail on how exactly they came up with these numbers, though I would highly encourage

you to look at paper if you're interested. I'll just mention the key takeaways. First, displaying good ads doesn't hurt publishers too much, in the sense that workers only chose to classify slightly fewer emails with good ads compared to no ads at all.

However, displaying annoying ads hurts a lot. Goldstein et al. estimated that they would have to pay about $1 extra to generate 1000 views using a bad ad compared with a good ad or no ad, which is extremely expensive for banner ads. In other words, publishers are likely losing money by displaying annoying ads unless they are charging these advertisers significantly more.

3 The Crowd Is Made of People

Bear with me while I take a moment to state the obvious: You know this "crowd" that we've been discussing? It's made up of *people*. And that's actually important to understand if you want to incorporate crowdsourcing into your own research.

Suppose that you would like to build your own hybrid intelligence system. Traditional computer science techniques allow us to reason about the performance of traditional computer systems. But what happens when the computer system is augmented with human computation? How can we analyze running time, scalability, or correctness? How do we predict the impact of our design decisions or optimize the system's performance when there are humans in the loop?

In order to do this, we need to have in mind a model of how the humans in the system are likely to behave. Are they mostly honest or will they cheat you if they can? Do they respond rationally to financial incentives? If we make unrealistic assumptions about the people who make up this system, then the system might not behave as we intended, so it's crucial to base our models on how people actually behave.[7]

What if you want to use crowdsourcing to generate training data, evaluate your machine learning model, or debug a machine learning system? Understanding the crowd is important here too. It can help you answer questions like how much you should pay for your task and how much you need to worry about spam. As we'll see, it also has implications about the independence of crowdworkers' responses, something that it's crucial to think about when studying human behavior too.

More broadly, understanding the crowd can help anyone who wants to use crowdsourcing do so more effectively. It can teach you how to set up the right payment structure for your tasks, how and why to communicate effectively, how to attract more people to your tasks, and how to avoid all sorts of common pitfalls.

In the second half of this tutorial, I will explore a series of studies aimed at understanding who the crowd is, what motivates them, and how they behave. Taken together, the results of these studies yield recommendations of best practices to follow when working with the crowd.

While many of the takeaways and lessons learned apply equally well to other crowdsourcing platforms,[8] most of the research I'll discuss in this section is looking specifically at workers on one particular platform,

[7] This is highly related to an agenda that my colleagues and I have been pushing within the theory community for the last few years. If you're interested, take a look at a review article on the topic which appears in this month's CACM [11] or at the websites of the recent CCC Workshop on Theoretical Foundations for Social Computing and HCOMP Workshop on Mathematical Foundations of Human Computation.

[8] https://www.quora.com/Are-there-any-similar-services-to-Amazon-Mechanical-Turk

Amazon Mechanical Turk. Amazon Mechanical Turk is a platform for crowdsourcing *microtasks* that has become popular within the research community. On Mechanical Turk, task requesters post small-scale *human intelligence tasks* (referred to within the Turk community as "HITs") along with the amount of money that they are willing to pay. Workers can then browse the set of tasks available and choose the tasks they want to work on.

3.1 Crowdworker Demographics

Over the years there have been several studies published which examine the demographics of workers on Mechanical Turk. I'm not going to spend a lot of time talking about these, but will mention a couple of statistics to give you a sense of the worker pool. These come from MTurk Tracker,[9] a project aimed at tracking the demographics of Mechanical Turk over time by continually releasing tasks containing demographic surveys on Mechanical Turk to obtain up-to-date information about workers [13]. While there are some potential problems with this approach (for instance, not all workers on Mechanical Turk choose to do surveys, so this is perhaps a better reflection of the population of workers who do survey work), the results are more or less in line with other studies.

According to the MTurk Tracker data:

- Somewhere around 70-80% of Mechanical Turk workers are from the United States, while about 10-20% are from India, but the breakdown of workers varies significantly throughout the day. The prevalence of workers from the U.S. and India makes sense because Mechanical Turk offers payment only in U.S. dollars, Indian rupees, or Amazon credit.
- The breakdown between male and female workers is fairly close to even, though it varies a bit by country.
- For crowdworkers in the U.S., the (self-reported) median household income is in the range of $40K-$60K, which is in line with the median U.S. household income. The median for Indian workers is less than $15K, with many Indian workers reporting a household income of less than $10K per year.

3.2 Spammers Aren't That Big of a Problem

A typical machine learning paper on the topic of aggregating labels from a crowd most likely includes a couple of reasons why the labels collected might be incorrect. In particular, it probably mentions the common notion that crowdsourcing platforms are riddled with spammers who try to cheat the system to make money. Do spammers and bots exist on real crowdsourcing platforms? Of course they do; this is the Internet after all. But most evidence suggests that they are not nearly as widespread as one might think and that the majority of crowdworkers are trying to do good work.

Sid Suri and colleagues [61] ran a study a few years back to test how honest the population of crowdworkers on Mechanical Turk actually are. Their study used a trick from the behavioral economics literature that allowed them to measure how trustworthy workers are on the whole even without being able to detect individual lies.

[9] http://www.behind-the-enemy-lines.com/2015/04/demographics-of-mechanical-turk-now.html

The idea is simple. Each worker was asked to roll a die (or simulate rolling a die on an external website) and report the value of her roll, a random number between 1 and 6. For completing this task, the worker received a base payment of $0.25 plus a bonus of $0.25 times her roll. For example, if a worker reported rolling a 4, she would receive a total payment of $1.25. Total payments were therefore between $0.50 and $1.75. Workers knew that there was no way for the requester to verify the outcomes of their rolls.

If all workers followed the instructions and honestly reported their rolls, we'd expect the mean of the rolls reported to be close to 3.5. What Suri and his colleagues observed was not so far off from this. The mean of the rolls reported by the 175 participants who took part in the study was 3.91. It appears that on the whole, workers had a tendency to overreport rolls of 5 and 6 and underreport rolls of 1 and 2, but this misreporting was far from universal, even in this extreme case in which workers had a risk-free way to directly benefit from being dishonest.

The authors of this work also tested whether increasing the ability of the requester to detect cheating would lead to higher levels of honesty. They conducted a variant of the die rolling experiment in which each worker was asked to report the results of thirty independent die rolls. This time, each worker received a base payment of $0.25 plus the sum of their reported die rolls, for a total payment between $0.55 and $2.05.

This time around, the mean reported role of the 232 participants was 3.57—still statistically significantly higher than the expected mean of 3.5, but significantly closer too. Examining the individual behavior of the subjects, only 3 of the 232 were found to have an average report significantly greater than we'd expect if they were honest, and only 1 chose to maximally game the system by always reporting 6.

These results are encouraging overall, but even if malicious workers are rare, it is still necessary to take precautions to avoid being scammed. Despite our best efforts, when we ran the communication network experiment I'll discuss a little later, we ended up paying one particularly devious worker more than six hundred times after he or she devised a complicated trick to exploit a bug in our system in order to submit work more than once. Mason and Suri [44] mention a similar incident.

Takeaways and Related Best Practices:

- Most workers are honest most of the time. Most of them are trying to do a good job.
- Despite this, you should still use care to avoid attacks. Even if most workers are honest, this is the Internet and scammers do exist.

3.3 Monetary Incentives

Next we'll turn our attention to the question of what motivates crowdworkers, and how both monetary incentives and intrinsic motivation can be used to improve the quantity and quality of crowdwork.

When researchers first started incorporating crowdsourcing into their work, a big part of the appeal was access to inexpensive data. Over time, the general viewpoint on this has shifted a bit. More emphasis has been placed on the ethical considerations of online labor markets and the importance of paying a fair wage. Many crowdworkers rely on the money they earn on Mechanical Turk to make ends meet. While crowdworkers are considered contractors and therefore are not covered by minimum wage laws, paying at

least minimum wage is the decent thing to do. It is good for you as a requester too, since it helps you maintain better relationships with workers, a point we will return to later.

When I've asked my colleagues in the human computation community how to set payments, I've heard the following tip: Ask your colleagues or students to complete your task or give your task to a small number of crowdworkers in order to calculate an estimate of how long it takes to complete. Use that estimate to make sure that workers receive the equivalent of the United States minimum wage (or higher).

Beyond being the decent thing to do, it is natural to ask whether paying more can improve the quality of crowdwork. I'll talk a bit about some of my own research here, focusing specifically on the impact of *performance-based payments* on the quality of crowdwork [24]. Performance-based payments are payments that reward crowdworkers for higher quality work. Most commonly, a worker is offered some base payment (in the examples I discuss, this will be $0.50, which we chose using the technique I mentioned above) just for completing a task with the opportunity to earn a bonus payment (say, an additional $1) for submitting work that the requester judges to be good. These types of bonuses are fairly common on platforms like Mechanical Turk.

When my collaborators and I began investigating performance-based payments, the literature on the effects of payments in crowdsourcing markets was a bit confusing and in some cases seemed to contradict itself. There was work showing that paying higher amounts increases the quantity of crowdsourced work, but not the quality [6, 41, 45, 54]. There was work showing that performance-based payments improve quality [22, 69] and other work showing that performance-based payments *do not* improve quality [57]. Finally there was work suggesting, perhaps surprisingly, that when performance-based payments are used, the quality of work does not depend on the size of bonus payments [68].

To make sense of and expand on these results, we set out to run a sequence of experiments with the goal of uncovering when, why, and on which tasks performance-based payments lead to higher quality. I'll describe a few of our experiments and the main take-away messages from our work. I'm going to spend a bit of extra time on this example because, in addition to telling us something about worker incentives, it also serves as a more detailed example of a behavioral experiment run on Mechanical Turk.

The first experiment we ran was a warm-up experiment to verify for ourselves that performance-based payments can indeed lead to higher-quality crowdwork on some task. At the same time, we wanted to test a hypothesis we had that even when a requester is not explicitly offering a bonus for high quality work, there may be a kind of *implicit* performance-based payment effect on workers. What do I mean by this? Well, on Mechanical Turk, work is not automatically accepted. While this doesn't often occur, a requester always has the opportunity to reject poor quality work and refuse payment. If a worker thinks her work is likely to be rejected if it is not high enough quality, then she may act as if payments are performance-based even if they're not.

For this experiment, we posted a proofreading task. We showed workers a block of text in which we had inserted a total of 20 common English-language typos. The block of text was an image so workers couldn't simply copy it and run it through a spellcheck. We chose this task because it had a few useful properties. First, quality was easily measurable since we inserted the typos ourselves. Second, we suspected that if workers exerted more effort it would lead to better results, which we guessed would make performance-based payments more effective. (I'll say more about this in a moment.)

We offered a base payment of $0.50 and a bonus of $1. We considered three bonus treatments: one in which

there was no bonus and no mention of a bonus, one in which all workers automatically received the bonus just for accepting the task, and one in which workers received the bonus only if they found at least 75% of the typos found by other workers. We chose to use this type of relative threshold for the bonus because it is something that could be implemented in practice without needing to know the total number of typos that could be found. Additionally, we did not want to make it obvious to workers that we had inserted the typos into the text ourselves and therefore knew how many there were. We chose an automatic bonus instead of a larger base payment so that we could post the task only once with a fixed base of $0.50 and randomly assign treatments after workers accepted the task.

To test the implicit performance-based pay hypothesis, we also considered two base treatments: one in which we explicitly guaranteed that we would accept a worker's work as long as she found at least one typo, and a treatment in which no such guarantee was mentioned.

Our results have a few takeaways. First, as you might expect, guaranteeing payment hurts. In other words, the implicit performance-based payment effect is real. Second, performance-based payments do indeed improve performance on this task. Finally, on this task, simply paying more (that is, giving a bonus independent of quality, just for accepting the task) also improves the quality of work. This is somewhat surprising at it contradicts what was observed in prior experiments. I will note, though, that while it led to a similar improvement in quality, giving unconditional bonuses costs the requester a lot more than using performance-based pay since the bonuses are awarded to everyone.

The next few experiments we ran were meant to test how sensitive this initial experiment was to the choice of task and to the particular parameters we chose (in this case, payments and thresholds).

First, we asked whether the results are robust to different choices of threshold. For this experiment, we varied what workers needed to do in order to receive the bonus payment. Our control had no bonus. We considered treatments in which workers needed to find either 25%, 75%, or all of the typos found by other workers to receive the bonus. Finally, we considered a treatment in which workers needed to find at least 5 typos total to receive the bonus.

There are a couple of interesting things to note. First performance-based payments led to high quality work for a wide range of thresholds. This is good news since it means the quality improvement is not too sensitive. Quality was slightly worse when we asked workers to identify all of the typos found by other workers, perhaps because they were less confident that they would be able to achieve this goal and gave up. Interestingly, it seems that making the threshold for payment uncertain leads to a big improvement. Since we inserted 20 typos, 25% of the typos would be 5, but quality was much higher when we asked workers to find 25%.

We additionally tested the effect of the bonus size. We found that as long as we offered a bonus that was big enough, quality improved. Offering a very small bonus (in this case $0.05) actually led to a small apparent decrease in performance, though this decrease is not statistically significant.

These results are especially interesting because they may explain some of the disparities in prior work. The paper of Shaw et al. [57] that claimed that performance-based payments don't improve quality used bonus payments that were extremely small compared with the base, so perhaps they were just in the regime where the payments were too small to help. On the other hand, the paper of Yin et al. [68] that noted that bonus sizes don't matter only considered bonuses that were relatively large compared with the base. We find, too, that in this regime, there are not statistically significant differences in quality when we vary the bonus size.

Finally, we wanted to understand what types of tasks are amenable to improvement from performance-based payments. In particular, we wanted to test our theory that these payments work well on *effort-responsive* tasks for which putting in more effort leads to higher quality. Surprisingly, when we tried to guess in advance which tasks would be effort-responsive, we were often wrong. To measure this more objectively, we had workers perform various tasks and looked at the relationship between the time it took each worker to complete the task and the worker's quality. We found that tasks like proofreading and spotting differences in images were effort-responsive, while handwriting recognition and audio transcription were not. Additionally, our experiments revealed that performance-based payments led to improved quality on proofreading and spotting differences, but not the others. This suggests that whether a task is effort-responsive may indeed play a role in whether quality can be improved using performance-based pay.

Takeaways and Related Best Practices:

- To be fair to workers, aim to pay at least the U.S. minimum wage. To figure out how much to pay, pilot your task to get a sense of how long it takes to complete. Paying higher than minimum wage can improve your relationship with workers.
- Performance-based payments can improve quality for effort-responsive tasks. If you aren't sure if your task is effort-responsive, try running a pilot to check the relationship between the time that workers spend on the task and the quality of their work.
- To be effective, bonus payments should be large relative to the base payment. As long as your bonus payment is large enough to make the reward salient, the precise amount doesn't matter too much, nor does the precise quality threshold a worker must meet to receive the bonus.

3.3.1 Intrinsic Motivation

Although Mechanical Turk is a paid crowdsourcing system, there have been several studies examining the effect of intrinsic, non-monetary sources of motivation for crowdworkers who use the platform.

Chandler and Kapelner [8] showed that workers are more active when tasks are framed as meaningful. They recruited workers on Mechanical Turk to label medical images. In one treatment, workers were told that they were labeling tumor cells and that the results of their work would be used to assist medical researchers. In the control, they were given no context for the task at all. In a third treatment, they were given no context and additionally told that the labels they generated would not be recorded; that is, all of their work would be discarded.

They found that when workers were told their work would benefit medical research, the quantity of work that they produced increased compared with the control, but their work was not significantly more accurate. On the other hand, when workers were told their work would be discarded, the quality of their work was worse than the control, but the quantity of work produced was similar.

Similar effects were observed by Rogstadius et al. [54] who compared the behavior of workers who were told they were performing work for a nonprofit organization "dedicated to saving lives by improving health throughout the world" with workers told they were working for a for-profit pharmaceutical manufacturer.

And of course, beyond paid crowdsourcing systems, the motivation to engage in meaningful work has been a major driver in the success of citizen science platforms and other volunteer-based crowdsourcing systems like the Zooniverse[10] and Science at Home[11].

Recently, Edith Law, Ming Yin, and collaborators [39] examined the possibility of appealing to workers' curiosity as a source of intrinsic motivation. Their work was inspired by the *information gap theory* of curiosity, which suggests that when people are made aware that there is a gap in their knowledge, they actively seek out the information needed to fill in this gap. They suggested several "curiosity interventions" aimed at stoking workers' curiosity. While some interventions increased worker productivity, there is some subtlety in how to most effectively engage workers' curiosity.

While I won't go into detail, gamification [14, 66] has also proved useful as a source of intrinsic motivation in both paid and unpaid crowdsourcing settings.

Takeaways and Related Best Practices:

- Crowdworkers produce more work when they know they are performing a meaningful task, but the quality of their work might not improve.
- Gamification and attempts to stoke curiosity can also increase worker productivity.

3.4 The Communication Network Within The Crowd

There is another assumption about the crowd that people often make without even realizing it: that crowdworkers are independent. When researchers post a task on Mechanical Turk, they expect the responses they receive from different workers to be uncorrelated, or even i.i.d.

Recent ethnographic studies have shown that this is not the case [19, 20]. Extensive interviews with crowdworkers have uncovered that it is common for workers to help each other with administrative overhead (especially in India, where even figuring out how to receive payment can be nontrivial), share information about good tasks and reputable (or irreputable) task requesters, and more generally recreate the social connections and social support that are missing from crowdwork. In other words, there is a hidden communication network behind websites like Mechanical Turk.

Last year, my collaborators Ming Yin, Sid Suri, Mary Gray, and I set out to quantify this hidden network in order to better understand the scale and structure of the network and how is it used, focusing on Mechanical Turk [70]. This is challenging because this network is not something that is accessible from an API or easily scrapeable. A lot of the communication goes on offline, for example, through text messaging or even in person. We needed to devise a way to map the network that would elicit as many "true" edges as possible and avoid eliciting edges that are not real (meaning it would probably not be a good idea to just pay per edge reported). We also needed a way to preserve workers' privacy. This meant we could not simply ask workers to report other workers' Mechanical Turk IDs since Turk IDs are not anonymous [40].

[10] https://www.zooniverse.org
[11] https://www.scienceathome.org

To do this, we created a web app, which we posted as a task on Mechanical Turk. When a worker accepted the task, he was first asked to create a nickname for himself. He then filled out a brief demographic survey and answered a couple of free-form questions about his experience on Mechanical Turk. These questions were carefully selected based on the results of a pilot study in which we asked workers what they were most interested in knowing about other workers on Mechanical Turk. We then asked the worker to pause and swap nicknames with other workers he knows who had already completed the task or might be interested in completing it. This process of swapping nicknames was how we constructed the network of communication. Workers were also able to return and add more nicknames later. When a worker added a connection to another worker, he was asked a few questions like how he usually communicates with this worker and what they communicate about.

Finally, we gave the worker a chance to explore the partially constructed worker network, viewing the network structure, basic information on all workers (including their answers to the questions from the pilot), and more extensive information about those workers with whom they had exchanged nicknames.

So what does the worker network look like? In the time that we ran the experiment, 10,354 workers completed our task. Based on previous estimates of the number of active workers on Mechanical Turk during any given period of time, we believe this is roughly a census of the worker population during that period [60]. These workers reported a total of 5,268 connections.

Roughly 13% of workers were connected to at least one other worker. On average these workers had 7.6 connections, and the maximum degree of any worker was 321. The largest connected component contained 994 workers, or about 72% of connected workers.

While workers reported communicating in many different ways, we found that the network was primarily enabled by the use of forums. 90% of all edges were between pairs of workers who communicate via forums, and 86% are between pairs who communicate exclusively through forums. These forums create visible subcommunities in the network. Our analysis showed that these subcommunities differ in terms of topological structure, dynamics, and the content of communication, with some acting more as social communities and others more like broadcasting platforms.

We found that connected workers tended to find our task earlier. Additionally, connected workers were more likely to have been active on Mechanical Turk longer and more likely to have achieved Mechanical Turk's Master level qualification. They also had a higher approval rate on average. While our experiment was not sufficient to show any causal relationship between connectivity and success on Mechanical Turk, it is consistent with the possibility that being connected has informational advantages to workers.

Takeaways and Related Best Practices:

- Forum usage is widespread on Mechanical Turk. You can think of forums as the virtual equivalent of a "water cooler" for crowdworkers. Workers go to these forums to share information about good and bad tasks and requesters.
- Engage with workers on forums. If you are a new requester, introduce yourself to the workers before you post your first task. Even if you're not new to Turk, posting information about your tasks on the forums can be a good recruiting tool. (We posted a notification on the forum TurkerNation when we were ready to launch the preliminary trial run of our network experiment in order to recruit a first

batch of workers who would be likely to know each other and therefore add links.)

- Actively monitor forum discussion of your task. We know that workers discuss tasks on forums. For some tasks, this can be beneficial; workers might share tips to help others complete your tasks more efficiently or accurately. In other cases, however, this discussion can be a big problem. This is particularly true if you are running a behavioral experiment with several different treatments. In these cases, including a polite request to avoid talking about your task in the task instructions or as part of the exit interview can be extremely effective, especially if you explain why. However, it is inevitable that someone will mention your task on the forums anyway, in which case you want to catch this quickly and shut the conversation down.
- Be careful about assuming independence. The workers who complete your task are not an i.i.d. sample of all workers on Mechanical Turk. For many applications, this is not a problem. But when coming up with your research methodology, make sure that you aren't implicitly assuming independence in a way that matters.

3.5 Additional Best Practices

I want to close by mentioning a few additional best practices that are crucial for effectively using Mechanical Turk for research, yet are rarely (if ever) mentioned in the literature. A few are covered in the survey article of Mason and Suri [44], while some are simply folklore in the crowdsourcing community. Some of these I learned through experience, and some were passed down to me by Sid Suri and others over the course of our collaborations. All are a consequence of our common refrain that *the crowd is made of people*.

The first few tips have to do with cultivating a good relationship with crowdworkers and building a good reputation for yourself as a requester. There are several reasons why this is important. We've already discussed how crowdworkers talk about good and bad requesters on forums. It is also common for crowdworkers to use tools that allow them to do things like view a requester rating when viewing a task or be notified when a favorite requester posts a task. Maintaining a good reputation leads to higher interest in your tasks.

- Actively monitor your requester email account and respond to questions. In deciding when to launch your task, make sure you will be able to set aside enough time to communicate with workers as needed. This takes some advanced planning, but it is worth it.
- Approve work quickly. Workers are not paid until their work is approved. Approving work quickly goes a long way towards maintaining a good relationship with workers.
- Avoid rejecting work. Maintaining a high approval rate is very important to workers. Many requesters only allow workers with sufficiently high approval rates to complete their tasks. Rejecting work from a well-meaning worker can therefore harm that worker's chance of earning future income. In general, work should be rejected only in the most extreme circumstances.

Finally, I'll mention a few more tips to help your project run smoothly.

- Pilot, pilot, pilot! No matter how carefully you think through the design of your task, your first implementation probably will not be perfect, especially if you're doing something novel. Run pilot

studies on your project collaborators, on your colleagues or students who are not directly involved in your project, and eventually, on small batches of crowdworkers. (If running an experiment, make sure to exclude these workers from future iterations of the task.)

- Iterate as many times as needed. It can be time consuming, but it is much better to catch bugs early rather than discovering them after you've fully launched your task.
- Create clear instructions. Pilot your task to collect feedback to make sure that your instructions can be understood. In some cases, it can make sense to insert quiz questions in the instructions to test worker comprehension and correct any misunderstandings.
- Create an attractive and easy-to-use interface. This is crucial for both keeping workers engaged and reducing errors from misunderstandings. Use pilots to test your interface too.
- When appropriate, conduct exit surveys for workers who have completed your task. Find out whether the instructions were clear, how they approached the task, and if they ran into any potential bugs or other issues.

For more tips on using Mechanical Turk in your research, take a look at the excellent survey article by Winter Mason and Sid Suri [44] and the references within. Although this article is targeted at researchers who wish to use Mechanical Turk for behavioral research, it should be required reading for any task requester. If you're interested in getting a workers' perspective, long-time crowdworker and crowd advocate Kristy Milland has several related articles and some useful slides from her guest lecture at Penn.[12] Finally, check out the guidelines for academic requesters[13] that were posted as part of the Dynamo project [55].

Acknowledgments

This tutorial was truly a crowdsourced effort! Thanks to all of the people—far too many to name—who sent me pointers and suggestions of exciting research to include. Thanks to Dan Goldstein, Chien-Ju Ho, Jake Hofman, Andrew Mao, Roozbeh Mottaghi, Sid Suri, Jaime Teevan, Ming Yin, Haoqi Zhang, and all of their collaborators for allowing me to borrow material from their slides for my presentation. Huge thanks to Chien-Ju Ho, Andrew Mao, Joelle Pineau, Sid Suri, Hanna Wallach, and especially Ming Yin for extended discussions and valuable feedback on prior versions of these notes. You guys are awesome! And thanks one more time to Sid, for passing down a lot of these folklore best practices for crowdsourcing over the course of our collaborations.

References

[1] Jacob Abernethy, Yiling Chen, and Jennifer Wortman Vaughan. Efficient market making via convex optimization, and a connection to online learning. *ACM Transactions on Economics and Computation*, 1(2):Article 12, 2013.

[12] http://crowdsourcing-class.org/slides/best-practices-of-best-requesters.pdf
[13] http://wiki.wearedynamo.org/index.php/Guidelines_for_Academic_Requesters

[2] Omar Alonso. Implementing crowdsourcing-based relevance experimentation: An industrial perspective. *Information Retrieval*, 16(2):101–120, 2013.

[3] Paul André, Haoqi Zhang, Juho Kim, Lydia B. Chilton, Steven P. Dow, and Robert C. Miller. Community clustering: Leveraging an academic crowd to form coherent conference sessions. In *HCOMP*, 2013.

[4] Pablo J. Barrio, Daniel G. Goldstein, and Jake M. Hofman. Improving comprehension of numbers in the news. In *CHI*, 2016.

[5] Michael Bernstein, Greg Little, Rob Miller, Bjoern Hartmann, Mark Ackerman, David Karger, David Crowell, and Katrina Panovich. Soylent: A word processor with a crowd inside. In *UIST*, 2010.

[6] Michael Buhrmester, Tracy Kwang, and Samuel D. Gosling. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 2011.

[7] Chris Callison-Burch. Fast, cheap, and creative: Evaluating translation quality using Amazon's Mechanical Turk. In *EMNLP*, 2009.

[8] Dana Chandler and Adam Kapelner. Breaking monotony with meaning: Motivation in crowdsourcing markets. *Journal of Economic Behavior and Organization*, 90:123–133, 2013.

[9] Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *NIPS*, 2009.

[10] Yiling Chen, Lance Fortnow, Nicolas Lambert, David Pennock, and Jennifer Wortman Vaughan. Complexity of combinatorial market makers. In *ACM EC*, 2008.

[11] Yiling Chen, Arpita Ghosh, Michael Kearns, Tim Roughgarden, and Jennifer Wortman Vaughan. Mathematical foundations of social computing. *Communications of the ACM*, 59(12):102–108, December 2016.

[12] Lydia Chilton, Juho Kim, Paul André, Felicia Cordeiro, James Landay, Dan Weld, Steven P. Dow, Robert C. Miller, and Haoqi Zhang. Frenzy: Collaborative data organization for creating conference sessions. In *CHI*, 2014.

[13] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G. Ipeirotis, and Philippe Cudré-Mauroux. The dynamics of micro-task crowdsourcing: The case of Amazon MTurk. In *WWW*, 2015.

[14] Oluwaseyi Feyisetan, Elena Simperl, Max Van Kleek, and Nigel Shadbolt. Improving paid microtasks through gamification and adaptive furtherance incentives. In *WWW*, 2015.

[15] Arpita Ghosh, Satyen Kale, and Preston McAfee. Who moderates the moderators? Crowdsourcing abuse detection in user-generated content. In *ACM EC*, 2011.

[16] Daniel G. Goldstein, R. Preston McAfee, and Siddharth Suri. The cost of annoying ads. In *WWW*, 2013.

[17] Daniel G. Goldstein, Siddharth Suri, R. Preston McAfee, Matthew Ekstrand-Abueg, and Fernando Diaz. The economic and cognitive costs of annoying display advertisements. *Journal of Marketing Research*, 51(6):742–752, 2014.

[18] Ryan Gomes, Peter Welinder, Andreas Krause, and Pietro Perona. Crowdclustering. In *NIPS*, 2011.

[19] Mary L. Gray, Siddharth Suri, Syed Shoaib Ali, and Deepti Kulkarni. The crowd is a collaborative network. In *CSCW*, 2016.

[20] Neha Gupta, David Martin, Benjamin V. Hanrahan, and Jacki O'Neil. Turk-life in India. In *The Intnernational Conference on Supporting Groupwork*, 2014.

[21] R. Hanson. Combinatorial information market design. *Information Systems Frontiers*, (1):105–119, 2003.

[22] Christopher G. Harris. You're hired! An examination of crowdsourcing incentive models in human resource tasks. In *WSDM 2011 Workshop on Crwdsourcing for Search and Data Mining*, 2011.

[23] Chien-Ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. Adaptive task assignment for crowdsourced classification. In *ICML*, 2013.

[24] Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan. Incentivizing high quality crowdwork. In *WWW*, 2015.

[25] John J. Horton, David Rand, and Richard Zeckhauser. The online laboratory: Conducting experiments in a real labor market. *Experimental Economics*, 14(3):399–425, 2011.

[26] Yuening Hu, Jordan Boyd-Graber, Brianna Satinoff, and Alison Smith. Interactive topic modeling. *Machine Learning*, 95:423–469, 2014.

[27] Ece Kamar. Directions in hybrid intelligence: Complementing ai systems with human intelligence. Abstract for IJCAI Early Career Spotlight Track Talk, 2016.

[28] David Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *NIPS*, 2011.

[29] David Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research*, 62:1–24, 2014.

[30] Ashish Khetan and Sewoong Oh. Achieving budget-optimality with adaptive schemes in crowdsourcing. In *NIPS*, 2016.

[31] Joy Kim, Sarah Sterman, Allegra Argent Beal Cohen, and Michael S. Bernstein. Mechanical novel: Crowdsourcing complex work through reflection and revision. In *CSCW*, 2017.

[32] Juho Kim, Haoqi Zhang, Paul André, Lydia B. Chilton, Wendy Mackay, Michel Beaudouin-Lafon, Robert C. Miller, and Steven P. Dow. Cobi: A community-informed conference scheduling tool. In *UIST*, 2013.

[33] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E. Kraut. Crowdforge: Crowdsourcing complex work. In *UIST*, 2011.

[34] Adriana Kovashka, Olga Russakovsky, Li Fei-Fei, and Kristen Grauman. Crowdsourcing in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 2016 (To appear).

[35] Raja S. Kushalnagar, Walter S. Lasecki, and Jeffrey P. Bigham. A readability evaluation of real-time crowd captions in the classroom. In *ASSETS*, 2012.

[36] Walter S. Lasecki and Jeffrey P. Bigham. Online quality control for real-time crowd captioning. In *ASSETS*, 2012.

[37] Walter S. Lasecki, Christopher D. Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey P. Bigham. Real-time captioning by groups of non-experts. In *UIST*, 2012.

[38] Walter S. Lasecki, Christopher D. Miller, and Jeffrey P. Bigham. Warping time for more effective real-time crowdsourcing. In *CHI*, 2013.

[39] Edith Law, Ming Yin, Joslin Goh, Kevin Chen, Michael Terry, and Krzysztof Z. Gajos. Curiosity killed the cat, but makes crowdwork better. In *CHI*, 2016.

[40] Matthew Lease, Jessica Hullman, Jeffrey P. Bigham, Michael S. Bernstein, Juho Kim, Walter S. Lasecki, Saeideh Bakhshi, Tanushree Mitra, and Robert C. Miller. Mechanical Turk is not anonymous. In *Social Science Research Network (SSRN) Online*, 2013.

[41] Leib Litman, Jonathan Robinson, and Cheskie Rosenzweig. The relationship between motivation, monetary compensation, and data quality among US- and India-based workers on Mechanical Turk. *Behavioral Research Methods*, 47(2):519–528, 2014.

[42] Qiang Liu, Jian Peng, and Alexander Ihler. Variational inference for crowdsourcing. In *NIPS*, 2012.

[43] Andrew Mao, Lili Dworkin, Siddharth Suri, and Duncan J. Watts. Resilient cooperators stabilize long-run cooperation in the finitely repeated prisoners dilemma. *Nature Communications*, 2016 (To appear).

[44] Winter Mason and Siddharth Suri. Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*, 44(1):1–23, 2012.

[45] Winter Mason and Duncan J. Watts. Financial incentives and the "performance of crowds". In *HCOMP*, 2009.

[46] Roozbeh Mottaghi, Sanja Fidler, Jian Yao, Raquel Urtasun, and Devi Parikh. Analyzing semantic segmentation using hybrid human-machine CRFs. In *CVPR*, 2013.

[47] Roozbeh Mottaghi, Sanja Fidler, Alan Yuille, Raquel Urtasun, and Devi Parikh. Human-machine CRFs for identifying bottlenecks in scene understanding. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[48] Iftekhar Naim, Daniel Gildea, Walter Lasecki, and Jeffrey P. Bigham. Text alignment for real-time crowd captioning. In *NAACL*, 2013.

[49] Gabriele Paolacci, Jesse Chandler, and Panagiotis G. Ipeirotis. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making*, 5:411–419, 2010.

[50] Devi Parikh and C. Lawrence Zitnick. Human-debugging of machines. In *Second NIPS Workshop on Computational Social Science and the Wisdom of Crowds*, 2011.

[51] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012.

[52] Genevieve Patterson, Chen Xu, Hang Su, and James Hays. The sun attribute database: Beyond categories for deeper scene understanding. *International Journal of Computer Vision*, 108(1–2):59–81, 2014.

[53] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11:1297–1322, 2010.

[54] Jakob Rogstadius, Vassilis Kostakos, Aniket Kittur, Boris Smus, Jim Laredo, and Maja Vukovic. An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. In *ICWSM*, 2011.

[55] Niloufar Salehi, Lilly Irani, Michael Bernstein, Ali Alkhatib, Eva Ogbe, Kristy Milland, and Clickhappier. We are dynamo: Overcoming stalling and friction in collective action for crowd workers. In *CHI*, 2015.

[56] Niloufar Salehi, Jaime Teevan, Shamsi Iqbal, and Ece Kamar. Communicating context to the crowd for complex writing tasks. In *CSCW*, 2017.

[57] Aaron D. Shaw, John J. Horton, and Daniel L. Chen. Designing incentives for inexpert human raters. In *CSCW*, 2011.

[58] Victor Sheng, Foster Provost, and Panagiotis Ipeirotis. Get another label? Improving data quality using multiple, noisy labelers. In *KDD*, 2008.

[59] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *EMNLP*, 2008.

[60] Neil Stewart, Christoph Ungemach, Adam J. L. Harris, Daniel M. Bartels, Ben R. Newell, Gabriele Paolacci, and Jesse Chandler. The average laboratory samples a population of 7,300 Amazon Mechanical Turk workers. *Judgment and Decision Making*, September 2015.

[61] Siddharth Suri, Daniel Goldstein, and Winter Mason. Honesty in an online labor market. In *HCOMP*, 2011.

[62] Omer Tamuz, Ce Liu, Serge Belongie, Ohad Shamir, and Adam Kalai. Adaptively learning the crowd kernel. In *ICML*, 2011.

[63] Jaime Teevan, Daniel Libeling, and Walter Lasecki. Selfsourcing personal tasks. In *CHI*, 2014.

[64] Jaime Teevan, Shamsi Iqbal, and Curtis von Veh. Supporting collaborative writing with microtasks. In *CHI*, 2016.

[65] Blase Ur, Jonathan Bees, Sean M. Segreti, Lujo Bauer, and Lorrie Faith Cranor Nicolas Christin. Do users' perceptions of password security match reality? In *CHI*, 2016.

[66] Luis von Ahn and Laura Dabbish. General techniques for designing games with a purpose. *Communications of the ACM*, 51(8):58–67, August 2008.

[67] Peter Welinder, Steve Branson, Serge Belongie, and Perona Pietro. The multidimensional wisdom of crowds. In *NIPS*, 2010.

[68] Ming Yin, Yiling Chen, and Yu-An Sun. The effects of performance-contingent financial incentives in online labor markets. In *AAAI*, 2013.

[69] Ming Yin, Yiling Chen, and Yu-An Sun. Monetary interventions in crowdsourcing task switching. In *HCOMP*, 2014.

[70] Ming Yin, Mary L. Gray, Siddharth Suri, and Jennifer Wortman Vaughan. The communication network within the crowd. In *WWW*, 2016.

[71] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I. Jordan. Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. *Journal of Machine Learning Research*, 17(102):1–44, 2016.

[72] Dengyong Zhou, Sumit Basu, Yi Mao, and John Platt. Learning from the wisdom of crowds by minimax entropy. In *NIPS*, 2012.

[73] James Zou, Kamalika Chaudhuri, and Adam Tauman Kalai. Crowdsourcing feature discovery via adaptively chosen comparisons. In *HCOMP*, 2015.