# Optimizing Open-Ended Crowdsourcing: The Next Frontier in Crowdsourced Data Management

Aditya Parameswaran[†], Akash Das Sarma[⋆], Vipul Venkataraman[†]
[†]University of Illinois [⋆]Stanford University

**Abstract**

*Crowdsourcing is the* primary means to generate training data at scale*, and when combined with sophisticated machine learning algorithms, crowdsourcing is an enabler for a variety of emergent automated applications impacting all spheres of our lives. This paper surveys the emerging field of formally reasoning about and optimizing* open-ended crowdsourcing*, a* popular and crucially important, but severely understudied *class of crowdsourcing—the next frontier in crowdsourced data management. The underlying challenges include distilling the right answer when none of the workers agree with each other, teasing apart the various perspectives adopted by workers when answering tasks, and effectively selecting between the many open-ended operators appropriate for a problem. We describe the approaches that we've found to be effective for open-ended crowdsourcing, drawing from our experiences in this space.*

## 1 Introduction

We are on the cusp of a new data-enabled era, where machine learning algorithms, trained on large volumes of labeled training data, are enabling increasing automation in our daily lives—from driving, robotics, and manufacturing, to surveillance, medicine, and science; a recent New York Times article calls this *"a transformation that many believe will have a payoff on the scale of the personal computing industry or the commercial internet"* [1]. Although considerable effort has gone into the development of machine learning algorithms for these applications, the generation of labeled training datasets, done at scale using *crowdsourcing* [2]— while equally important [3, 4]—is often overlooked. Recent work has demonstrated that *optimizing crowdsourcing* can often yield *orders of magnitude more high-quality labeled data at the same cost*, spurring the development of increasingly sophisticated machine learning algorithms, and providing immediate benefits via substantial increases in accuracy for existing algorithms.

**From Boolean Crowdsourcing to Open-Ended Crowdsourcing.** However, most of the past research on optimizing crowdsourcing has focused on what we call *boolean crowdsourcing*, e.g., [5–9, 11, 47], where human involvement can be abstracted as tasks or operators whose answers come from a small, finite domain, e.g., evaluating a predicate, comparing a pair of items, or rating an item on a scale from 1–5. In the former two cases, the domain of possible answers is boolean, while in the last case, the domain is finite and has cardinality five. Boolean crowdsourcing operators have natural analogs in computer operators, and are easier to reason about and develop algorithms with. This research has largely ignored *open-ended crowdsourcing*, where the answers to


**Bulletin of the IEEE Computer Society Technical Committee on Data Engineering**

the tasks have no similar restriction. Anecdotal evidence indicates that open-ended crowdsourcing tasks are at least as popular as traditional boolean crowdsourcing tasks on crowdsourcing marketplaces. As one data point, an analysis of Mechanical Turk's log data [12] reported that *content creation*—including open-ended tasks like transcription—was by far the single largest category of tasks in the period from 2009 to 2014. As another example, a survey of industry users of crowdsourcing reports similar findings [2].

**Examples of Open-Ended Crowdsourcing.** We now describe some canonical open-ended crowdsourcing problems that we use as examples for this paper. One example is *transcription*: the goal of transcription is to transform a piece of audio or video to text. This could be done, for example, using unit tasks where we provide workers a portion of the audio or video, along with a text box where they can type a sequence of words to match the contents of the portion. Another example is *clustering*: the goal of clustering is to subdivide a collection of items (images, pieces of text) into clusters. This could be done, for example, using unit tasks where workers place items into an arbitrary number of buckets determined by them. Yet another example is *detection*: the goal of detection is to identify where in an image an object is located. This could be done, for example, using unit tasks where workers draw a bounding rectangle or polygon around the location of the object of interest.

**Open-Ended Crowdsourcing: A Pressing Need.** Beyond the popularity, there are several reasons why open-ended crowdsourcing is crucially important. First, some tasks are *near-impossible with just boolean crowdsourcing*. For example, using boolean crowdsourcing to locate the position of an object in an image is near-impossible. If we use a task like: "is the object in this portion of the image (yes/no)", we may be able to get close to the actual location of the object by repeatedly using this task on various portions of the image, but getting an accurate bounding box around the object may require hundreds or thousands of such tasks. On the other hand, asking workers to simply draw a bounding box is a lot more effective in terms of time, cost, and accuracy. Second, open-ended crowdsourcing lets us *get more fine-grained data*, since workers provide answers from a potentially unbounded set. If we were to use an entropy argument, the number of bits provided by workers for an open-ended question or task is considerably larger than that provided for a boolean question. Third, recent computer vision and text processing papers argue that *fine-grained training data is essential* for developing sophisticated machine learning models [13–16]. Specifically, our best hope for improving the accuracy of present machine learning models is by using training data that reveals more information about how humans think, as opposed to training data that is more akin to how computers operate (i.e., via boolean operations).

*Despite these compelling reasons, research on optimizing open-ended crowdsourcing is still in its infancy.* Open-ended crowdsourcing is presently *leveraged in an unoptimized fashion, with resources wasted and inaccurate data collected, or even worse, not at all,* leading to severe impediments to machine learning.

**Challenges.** The reason why open-ended crowdsourcing is leveraged in an unoptimized fashion is that optimizing open-ended crowdsourcing is substantially harder than optimizing boolean crowdsourcing:

*1. Hard to aggregate.* Due to the many possible answers to open-ended tasks, distilling the 'right' answer from this set is non-trivial. For example, when drawing a box around an object in an image, or transcribing audio into text, no two workers will provide the same box or transcription. This is because the number of possible answers is very large, and there are many ways of making mistakes. In boolean crowdsourcing, we could simply resort to the majority opinion, but those techniques do not apply, especially when all of the answers are different from each other.

*2. Sparsity of quality measurements.* In trying to characterize the error rates of workers, due to the large number of possible answers, it is hard to get reliable estimates of the probability that a worker provides an answer $a$, when the true answer is $b$. In order to estimate these probabilities, it would take a lot of tasks to be issued on crowdsourcing marketplaces to ensure adequate coverage (and therefore accurate estimates) for every $(a, b)$ pair.

*3. Many right answers.* To further complicate matters, open-ended tasks often have many 'right' answers, due to different underlying perspectives or beliefs, making it challenging to distinguish between the case when a worker is making mistakes, or the case when the worker is simply adopting a different perspective. For instance,

when clustering items—say a collection of images of everyday objects, workers may use different criteria—size, color, geometric shape—to cluster the items, while also inadvertently introducing errors.

*4. Multiple scales.* While boolean crowdsourcing typically operates on items (images, text) as a whole, open-ended crowdsourcing can additionally operate on portions of items. For example, when counting objects in an image, we may ask workers to count within portions of the image for less error-prone counts. There are an unbounded number of such portions that can be counted, making it hard to pick between them when selecting tasks to be assigned to workers.

*5. Many open-ended operators.* Unlike boolean crowdsourcing which is limited to a small number of operator types, there is a wide variety of open-ended operators, even for the same problem (including boolean ones as a subset). For example, for detecting where an object is present in an image, workers could draw a box around an object, fix a box, or compare boxes. The large number of alternatives makes it hard to design algorithms.

In short, these issues (1–3) lead to an increased complexity in reasoning about the underlying algorithms, and (4–5) an overwhelming number of design choices when it comes to designing algorithms.

**This Paper.** While there has been a large body of work on open-ended crowdsourcing, primarily from the HCI (Human-Computer Interaction) community, most of this work has been on creatively using open-ended crowdsourcing operators in workflows as opposed to understanding how to model and reason about them, and develop optimal algorithms. While there has been a deep, interesting, and important body of work from the database community (and to a certain extent from the AI and machine learning communities) on optimizing boolean crowdsourcing, our hope is to bring open-ended crowdsourcing the same kind of attention, especially given its importance as articulated above.

Therefore, in this paper, we aim to outline an emerging body of work in optimizing open-ended crowdsourcing from us and from other groups by developing a set of *design principles* that we have found to be effective for algorithm development, and by describing how these design principles were applied for a few papers that we have been working on in this space. Note that our survey of the emergent work on open-ended crowdsourcing will necessarily be biased by our own work that we're most familiar with; this is not to indicate that the other work is not as important or as interesting, but merely indicates our lack of familiarity with them. We will attempt to categorize all of the open-ended crowdsourcing work from the database community that we are aware of, along with work from other communities in Section 4.

## 2 Design Principles for Open-Ended Crowdsourcing

We now describe some approaches that we've found to be successful in dealing with the increased modeling complexity (issues 1–3 above) and increased number of design choices (issues 4–5 above), followed by a solution scaffold or recipe for open-ended crowdsourcing.

### 2.1 Dealing with the Modeling Complexity

The challenges in modeling worker performance stems from the fact that there are far too many possible answers that workers can provide even for a single question or task. This means that we do not have a clear mechanism to aggregate worker answers, and nor do we have the ability to estimate error rates of the form $\Pr[\text{worker answers } a|\text{true answer is } b]$ for all $(a, b)$ pairs. We have identified two ways of dealing with this modeling complexity, both of which essentially allow us to *transform* the worker answer into one or more boolean crowdsourcing answers, following which we can apply standard techniques from boolean crowdsourcing.

The first approach is to *project the answer down* to a finite set of choices. For example, if the worker provides an answer that is any rational number in a range, we can project this answer down to a finite set of integers; yet another way is to project it down into a binary choice: $\leq a$ or $> a$. Note that this approach is wasteful in that we lose some of the fine-grained information that workers are providing, and begs the question of why we

didn't simply ask the boolean crowdsourcing question in the first place. Nevertheless, this simple approach is commonly used: we provide an example of this approach in Section 3.1.

The second approach is to *decompose the answer* down to answers to a collection of boolean crowdsourcing questions. As an example, if a worker is providing a transcription to a piece of audio, instead of treating the entire transcription as one open-ended crowdsourcing task, we can break it down into the sequence of individual words, each of which can be considered a response or a non-response to a word transcription task, which is much more manageable. By doing so, we can now model and reason about workers making mistakes at the level of words, rather than complete transcriptions. In transcription, at least, an additional challenge remains, which is to identify which words across workers were meant to be provided as output for the same portion of the audio piece. As another example, if a worker is providing a bounding box as an answer to a detection task, instead of treating the bounding box as a whole, we can decompose it down into boolean crowdsourcing answers for individual pixels: where if a pixel is part of the box drawn by a worker, the pixel gets a "yes" answer for the corresponding boolean crowdsourcing question, while it gets a "no" answer if it is not. This approach has the downside that the answers to the "pixel-level" boolean crowdsourcing questions are in fact not independent—if the answer to a specific pixel is "yes", then it is more likely than the answer to a neighboring pixel is "yes" rather than no. By decomposing the open-ended crowdsourcing task to boolean crowdsourcing ones, we lose this information.

We also employ a third approach which is fundamentally different from the previous two. This last way of dealing with open-ended crowdsourcing tasks is a bit more ad-hoc, and problem dependent, but does not require us to discard any information. Here, we operate on the answers provided to the open-ended crowdsourcing tasks directly. Consider the answers to a single open-ended crowdsourcing task. We can represent each of these task answers as nodes in a graph, and connect nodes that are similar to each other (on some similarity metric, such as overlap) with an edge annotated by the degree of similarity. Once this graph of answers is constructed, we can apply standard graph clustering algorithms to identify various view-points among the open-ended task answers: the largest cluster or clique may represent the "consensus" answer. What can be done with these clusters is dependent on the problem. We provide an example of this approach in Section 3.2.

## 2.2 Dealing with the Increased Design Choices

As described previously, open-ended crowdsourcing brings with it a considerable increase in the number of alternative tasks that can be issued at each step. The increase is due to the large number of open-ended crowdsourcing task types available, and also because these tasks can be applied to portions of items and not just the items directly. We now describe our approaches to deal with these increased design choices.

Our first approach is one that has been applied in the past for boolean crowdsourcing, which is to estimate the *information gain* for issuing a specific task: here, an additional challenge is that the information gain may be hard to estimate because we do not have a good model to reason about worker performance. Nevertheless, we may be able to use proxies for information gain, e.g., prioritizing items or portions of items for which we have fewer answers than others, or more ambiguity, perhaps measured by projecting or decomposing the answer down to boolean tasks (as described in the previous section).

The second approach is to start at a coarser granularity and then drill-down only when needed. For instance, in transcription—we can have workers first transcribe the entire audio piece, and then have additional workers transcribe the portions that are were found to be ambiguous. We will describe another example of this approach in Section 3.1.

A last approach is to incorporate information from primitive machine learning algorithms to "direct" attention to specific items or portions of them. For example, if we know for certain that an object does not lie in a given portion of the image, we can remove those portions when providing workers the open-ended task where they are asked to draw a bounding box about the object of interest. Once again, we describe how this approach is used for counting in Section 3.1.

### 2.3 Solution Scaffold

We have developed an *open-ended crowdsourcing problem-solving approach* drawing on our mechanisms to deal with additional modeling and design choice complexity. While the specific instantiation may differ across problems, the overall principles still apply, and the insights transfer across. For each problem, we apply **MARQED**, short for ***Model-Aggregate-Reason-Quantify-Estimate-DrillDown***: the first three (MAR) are tailored to managing modeling complexity, while the last three (QED) are tailored to managing design choice complexity.

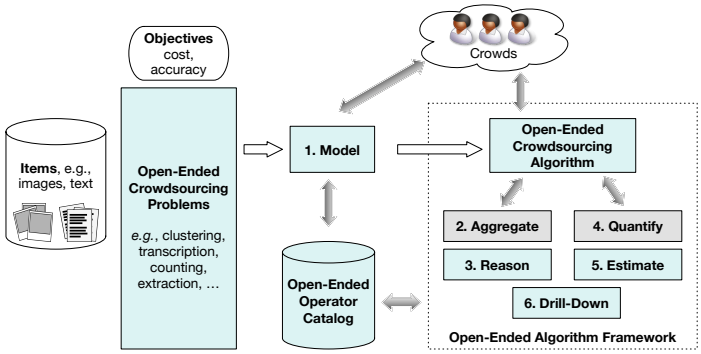


Figure 1: Algorithm Flow: Boxes in blue did not exist in boolean crowdsourcing, while boxes in gray are substantially different

In particular, **MARQED** stands for the following: (1) ***Model*** the performance of workers on open-ended operators; (2) Develop methods to ***Aggregate*** across their responses to identify one or more 'right answers'; (3) Identify techniques to ***Reason*** about whether workers are generating their answers based on the same or different underlying perspective; (4) Develop procedures to ***Quantify*** the information gain of different open-ended operators, and the same operator operating on different items; (5) Design schemes to incorporate prior ***Estimates*** from automated algorithms to reduce costs; and (6) Develop ***Drill-Down*** techniques on the items to enable workers to examine an item more closely. Then, we ***design the open-ended crowdsourcing algorithm*** that can leverage (1–6) (if available)—see Figure 1; boxes in blue did not exist in boolean crowdsourcing, while boxes in gray are substantially different. Note that it is not necessary to develop solutions for all of (2–6) before we can reap considerable benefits in practice. Next, we describe how we applied **MARQED** in practice.

## 3 Example Problems in Open-Ended Crowdsourcing

In this section, we describe our results for two problems that we believe are representative of the challenges in open-ended crowdsourcing, along with our solution approaches, in addition to other problems in this space.

### 3.1 Counting

The goal of *counting* is to estimate the number of objects of a given type in an image at low cost; it is a basic computer vision primitive, with applications in security, medicine, and biology. Counting is a hard problem due to occlusion—the partial obscuring of objects behind other objects, with state-of-the-art automated techniques having accuracies less than 50% [17, 18].

In our paper [19], we develop cost-effective techniques to use crowds to accurately count the number of objects in images from two completely different domains—cell colonies in microscope photos, and people in Flickr photos. We now describe the components of our solution, drawing parallels to the **MARQED** methodology.

**Model.** Our open-ended operator is simple—we display an image or a portion of an image, and ask workers to provide the count of the number of objects in that portion. We find that workers do not make mistakes in counting when the number of objects in the image is less than a certain number $k$ (20 in our experiments); after that, workers start introducing errors, with the errors growing superlinearly as the number of objects increases. This may be because workers are not able to keep track of the objects they have already counted, or because they get fatigued beyond a certain point and start introducing errors. Using this insight, we model worker error by *projecting down* worker answers to boolean ones: if the answer is $< k$, then it is assumed to be correct; if the

answer is $\geq k$, then it is assumed to be incorrect i.e., given a worker provides a count $\geq k$, the only information we can deduce is that it is $\geq k$, but no additional information can be inferred.

While this simple model provides reasonable results, we are indeed wasting information by ignoring worker answers if they are $\geq k$. Using a more fine-grained error model along with maximum-likelihood estimation can help identify a current best estimate for an image or a portion of an image, allowing us to "skip ahead" to the portions that need more attention. That said, this fine-grained error model requires more expensive training data to estimate accurately.

**Drill-down.** Based on the simple error model, we can already develop a strategy for counting objects in images, by repeatedly splitting images and *drilling-down*. We model this process as a *segmentation-tree*, where the root node represents the original, complete image, and children of any node represent the segments obtained by splitting the parent image (using some splitting scheme, horizontal or vertical). Figure 2 shows one such example segmentation tree where the original image, $V_0$, is split into segments $\{V_1, V_2\}$, which are respectively split into $\{V_3, V_4\}$ and $\{V_5, V_6, V_7\}$.

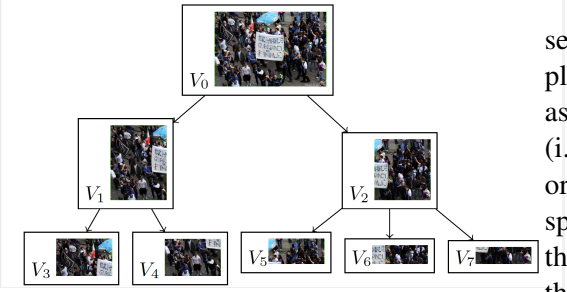

Figure 2: Segmentation Tree

We refer to any set of mutually exclusive nodes (i.e. image segments) that when put together reconstruct the original complete image as a *frontier* of the segmentation-tree. We start by asking workers to count the number of images in the root node (i.e. the complete image). If they reply with a count greater than or equal to $k$, then we traverse down the segmentation tree by splitting the image and asking workers to provide the count for the children segments. If the count on all segments is smaller than $k$, then we are done. If not, we again split and repeat this process for every segment that still has a count $\geq k$, until we reach a frontier of image segments such that every one of them has a smaller count than $k$. This simple strategy has optimal *competitive ratio* for any given segmentation tree, and also returns counts with reasonable accuracies of 93% when counting people on a standard dataset [20]. This approach does come with one challenge: objects often span multiple sibling segments in the segmentation tree, and therefore have a danger of being double-counted. In our paper, we take the easy-way-out by asking workers to only count an object if more than half of it appears in the image segment. Further improvements may be possible.

**Reason and Aggregate.** Using our simple error model, reasoning about perspectives, and aggregation is not necessary—workers are expected to agree and provide correct answers as long as the true count is $< k$—thus we can simply ask one worker to count each image or portion of the image, and additional answers are not necessary. In practice, however, we find that sometimes workers may still introduce errors: to handle this, in our paper, we take three worker answers per question and take the median—this simple aggregation scheme is sufficient to resolve worker differences and obtain high accuracy count estimates. Finally, we sum up all the aggregated counts from the different constituent image segments to obtain the count for the original, whole image. Changes to the worker error model, for instance, using a fine-grained probabilistic model for maximum-likelihood count estimation, will open the road to more interesting aggregation challenges.

**Quantify.** While we did not implement other operators for this problem, there are a number of interesting alternatives that yield more information, at the expense of a little extra worker effort. For instance, we could ask workers to tag objects that they have already counted, say with a dot, to help eliminate double-counting as well as avoid missing objects. Tagged objects could serve as references for other workers, who could then mark additional objects missed by previous workers, or eliminate redundant instances.

**Estimate.** Even if the number of objects is in the hundreds, our algorithm on the segmentation tree may end up asking several "useless" questions at the higher levels all with count $\geq k$, while the "useful" questions (whose answers are actually used to compute the final count at the root) are at the frontier of the tree where the count is just $k$. However, it may be possible to use feedback from a primitive automatic segmentation algorithm to craft

a segmentation tree where there are no useless levels, and where objects do not span multiple sibling segments. Consider the problem of counting cells in biological images. Even though automated counting may be hard due to occlusion, we can partition the image into non-overlapping portions using the watershed algorithm [21] and learn prior counts for each partition using an SVM [22]. Note that these prior counts may be much smaller than expected (due to occlusion), but it suffices as a starting point.

Given these partitions and prior counts, we can construct a segmentation tree that groups multiple contiguous partitions together until they hit up to $k$ objects each (based on the prior counts, which may be underestimating). This allows us to construct a segmentation tree where all levels are "useful" given our prior information. Since merging partitions together optimally is NP-HARD via a reduction from planar partitioning [23], we employ other heuristic techniques in our paper, such as first-fit [24]. We then traverse the tree asking workers as before. By using the prior machine-learned estimates in this fashion, we are able to skip several "useless" questions providing a 2x reduction in cost. It should be noted that the images of cell colonies are much more amenable to automated prior estimation techniques. It is an open challenge to explore whether similar techniques could be applied to other kinds of objects, where it is a lot harder to get accurate prior counts.

### 3.2 Clustering

Given a collection of items (e.g., images, documents), the goal of clustering, is to organize them into coherent groups. Collections of images or documents are commonplace; organizing them is essential before one can understand the themes in the collection, or improve search or browsing. Clustering embodies all the challenges faced by open-ended crowdsourcing: workers can have different perspectives leading to multiple "right" answers making worker responses hard to aggregate. The space of possible responses is also extremely large, since workers operate on multiple items simultaneously. The open-endedness of the problem also means that there are a number of different interfaces and operators that could conceivably be used. We shall now see how applying the **MARQED** approach allows us to reason about this challenging open problem in a principled fashion.

**Model.** Prior work has considered crowdsourced clustering, limited to the boolean operator where pairs of items are compared [25, 26]—as a result, crowd workers do not have any context to compare items. Also, the eventual clusterings end up having "mixed" perspectives, resulting in low accuracies. Instead we used a basic open-ended clustering operator, where a set of items are provided to workers, and they are asked to group them into an arbitrary number of disjoint clusters. Using this operator, we had multiple workers cluster a stylized image dataset with each image containing a shape with different sizes and colors, as a running example. First, we introduce the notion of *concept hierarchies* to capture the notion of worker cluster perspectives. One concept hierarchy for the concept of shapes, could be to have {*All Shapes*} divided into {*Quadrilaterals*, *non-Quadrilaterals*}, the latter of which is subdivided into {*Triangles*, *Circular Shapes*}. For any given dataset, the concept hierarchy need not be unique. For instance, another hierarchy would have {*All Shapes*} divided into {*Circular Shapes*, *Straight-Edged Shapes*}. When clustering, each worker answer can be seen to draw from one or more inherent concept hierarchies.

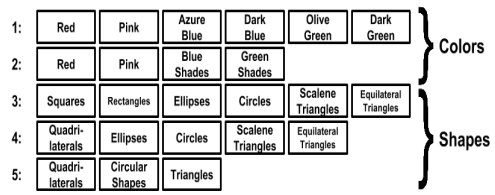

Figure 3: Examples of real worker clusterings

Figure 3 shows examples of real worker clusterings on our dataset. While workers 1 and 2 clustered based on color alone, workers 3, 4 and 5 clustered based on shape. We focus on the latter for the time being. One conceptual hierarchy, $C$, "consistent" with workers 3, 4, and 5 is {*All Shapes*} divided into {*Quadrilaterals*, *Triangles*, *Circular Shapes*}, which are respectively subdivided into {*Squares*, *Rectangles*}, {*Scalene Triangles*, *Equilateral Triangles*}, and {*Circles*, *Ellipses*}. We additionally introduce the notion of *frontiers* on a given concept hierarchy to capture the notion of granularities: a frontier in a hierarchy is a set of nodes that do not have any ancestor-descendent relationship between them, and together cover all paths to the leaves. Each frontier corresponds to one valid granularity of clustering consistent

with the concept hierarchy. Representing the concept hierarchy, $C$, as a tree, we have worker 3 operating at the leaf nodes, or at the finest granularity of the tree, corresponding to the frontier {*Squares*, *Rectangles*, *Scalene Triangles*, *Equilateral Triangles*, *Circles*, *Ellipses*}. Similarly, worker 5 is operating at a depth of one in the tree, which is the coarsest non-trivial granularity and corresponds to the frontier {*Quadrilaterals*, *Triangles*, *Circular Shapes*}.

**Reasoning and Aggregation.** From our experiments on the stylized dataset, we make the following observations: (a) Workers cluster using different *perspectives*, e.g., some workers clustered using shape, and others clustered using color. That said, there was a dominant, popular clustering perspective, in this case, shape. (b) Even within a perspective, workers cluster at various *granularities*, e.g., some workers clustered shapes into rectangles and non-rectangles, while others broke up non-rectangles into fine-grained clusters. (c) Sometimes, there are *confusing items* that end up being placed in different clusters by different workers even if they agree on the perspective and the granularity.

To reason about workers' perspectives, we develop a notion of *consistency*: two worker clusterings (i.e., a set of clusters formed by each worker) are consistent if for any pair of clusters $C, C'$, one from each worker, either $C \subseteq C', C' \subseteq C$, or $C \cap C' = \emptyset$, i.e., each cluster from one worker generalizes, specializes, or does not overlap with another worker's clusters. This definition allows consistent workers to cluster at different granularities.

Given the notion of consistency, we can now directly operate on worker responses to identify whether there are any "consensus" clusterings, i.e., consensus concept hierarchies (perspectives), and frontiers within them (granularities), that emerge. This allows us to have a starting point to cluster the rest of the items. To do this, we generate a *clustering graph*, with one node per worker, with consistent pairs having an edge between them.

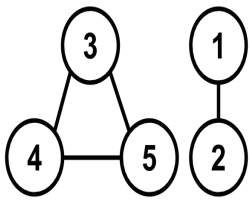

Figure 4: Clustering graph

Figure 4 shows the clustering graph corresponding to the workers from Figure 3. Workers 1 and 2 both clustered based on color alone and were consistent with each other, while workers 3, 4, and 5 clustered by shape alone with no inconsistencies among themselves. We can show that under multinomial worker perspective selection models, the maximum likelihood worker perspective is the MAX-CLIQUE in the graph. Despite MAX-CLIQUE being NP-HARD, the clustering graph is often not large, making the problem tractable. In our paper, we additionally describe how we can incorporate worker error models, which introduce additional complexity to the problem.

**Quantify and Drill-Down.** So far, we have described how to use a single open-ended clustering operator and aggregate responses from it for a small number of items. However, if we have a large number of items, workers might not be able to cluster all of them in one go. This suggests the need for drilling-down, or splitting the set of items, and asking workers to cluster the resulting subsets. This raises the challenge of aggregating partial clusterings. For this purpose, we maintain a *kernel* of items across clustering tasks, akin to *pivots* in Polychronopoulos et al. [9], to be able to relate partial clusterings across each other, for aggregation. We design techniques to extend the current maximum likelihood hierarchy by merging worker responses on new items to the existing hierarchy—we leverage these techniques to design a merging algorithm which aggregates clusterings from separate subsets together to output a consensus hierarchy on the original, complete set of items.

We additionally incorporate a categorization-based operator, once the consensus clustering is identified, to provide additional cost benefits, by categorizing the remaining items into the discovered clusters [27]. Instead of having workers repeatedly cluster many items (implicitly a many-to-many comparison), they only need to identify which cluster or category is appropriate for one item at a time, with the clusters being fixed.

Overall, our techniques lead to up to $3\times$ better recall and $1.9\times$ better accuracy than boolean clustering schemes using pairwise judgments [25, 26] on their datasets, for the same crowdsourcing cost.

### 3.3 Other Problems

We've applied the **MARQED** approach to other open-ended crowdsourcing problems. We describe some of these problems briefly here, followed by work done by others.

**Extraction.** The goal of *extraction* is to use crowdsourcing to gather entities of a specific type [28]. We considered a broad space of open-ended operators for this problem, leveraging the attributes of the entity set [29]. For example, musicians in Chicago can be categorized as guitarists, drummers, and so on, and may play music of various types. By considering these attributes, we can ask workers to answer more *fine-grained* open-ended questions: e.g., provide a jazz drummer in Chicago. This allows us to target the questions at the attribute combinations where we lack entities. However, the number of possible open-ended questions increases exponentially in the attributes, making the problem challenging. We showed that picking the best questions is NP-HARD, and found that a *drill-down* based technique works well, where we ask generic questions first (e.g., provide a musician), and then drill down and ask more specific questions later (e.g., provide a drummer in Lincoln Park).

**Searching.** The goal of crowd-assisted search [30, 31], is to return relevant results from a corpus given a search query with embedded images, video, and/or text. Here, we once again found that the following open-ended problem solving aspects are helpful: (a) multiple open-ended operators; (b) starting at a coarser granularity and drilling down into more promising candidates; and (c) incorporating prior information – in this case from regular text search engines.

**Batching.** Boolean tasks are typically grouped into batches of 10s to 100s of tasks that are attempted by workers together—to reduce cost and effort—making it essentially one large open-ended task. However, these tasks are assumed to be answered independently, which is not the case. We developed a probabilistic model to reason about the answers incorporating two forms of judging: independent, where workers answer each question independently, or correlated, where workers implicitly rank the items and then pick the top-$k$ to be positive, with the rest negative (i.e., the Plackett Luce model). We found that this model, led to substantial accuracy improvements over schemes that ignored these correlations [32].

In addition, there has been work by others on optimizing other open-ended crowdsourcing problems. We describe some of them below.

The goal of *transcription* is to transcribe a sequence of words into text; it is a very challenging problem, with automated techniques performing very poorly [33, 34]. By decomposing worker answers down into individual words, one can apply standard multiple sequence alignment algorithms from the bioinformatics literature to align worker answers and identify consensus answers, leading to substantial improvements [34]; other work tries combining crowd answers with automated techniques [35, 36]. Some prior work has also looked at the problem of *detection* — finding the location of objects in images, applying computer vision models and human input via a Markov Decision Process [37].

Other open-ended crowdsourcing work exists as well, for various purposes, including designing plans [38], rule mining [39], and pattern matching [40].

## 4 Related Work

The work on open-ended crowdsourcing is related to work in many areas.

**Area 1: Optimized Boolean Crowdsourcing.** There has been quite a bit of work on optimizing boolean crowdsourcing, targeting basic algorithms, including filtering [5,6,41], sorting [7], max [11,42,43], categorization [27], top-K [8, 9, 44], spatial crowdsourcing [45, 46], and entity resolution (ER) [47, 47–53].

**Area 2: Crowdsourcing Systems and Toolkits.** Many groups have been building crowdsourcing systems and toolkits to harness crowdsourcing in a "declarative" manner [54–57], as well as several domain-specific toolkits [30, 31, 58, 59]. All these systems and toolkits could benefit from the design of optimized algorithms as building blocks.

**Area 3: Quality Estimation:** A number of papers perform simultaneous worker quality estimation and most accurate answer estimation, typically using the EM algorithm, sometimes providing probabilistic or partial guarantees, and sometimes modeling difficulty, bias, and adversarial behavior, e.g., [60–67]. To our knowledge, there is no work on applying EM to open-ended crowdsourcing tasks.

**Area 4: Decision Theory:** Recent work has leveraged decision theory for improving cost and quality in simple workflows, typically using POMDPs (Partially Observable MDPs), to dynamically choose the best decision to make at any step, e.g.. [68, 69]. While some of this work could be applicable to some open-ended tasks, there are no optimality guarantees associated with any of these techniques.

# 5 Conclusion

Open ended crowdsourcing is not only challenging and interesting, but also necessary to meet the demands of the new generation of data-hungry applications. We hope that the next wave of crowdsourcing research from the database community will tackle more problems in this space, expanding the frontier of our understanding.

**Acknowledgements:** A. P. acknowledges support from grants IIS-1513407 and IIS-1633755 awarded by the National Science Foundation, grant 1U54GM114838 awarded by NIGMS and 3U54EB020406-02S1 awarded by NIBIB through funds provided by the trans-NIH Big Data to Knowledge (BD2K) 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 and organizations.

# References

[1] "Artificial Intelligence Swarms Silicon Valley on Wings and Wheels," *http://www.nytimes.com/2016/07/18/technology/on-wheels-and-wings-artificial-intelligence-swarms-silicon-valley.html*.

[2] A. Marcus and A. Parameswaran, "Crowdsourced data management: Industry and academic perspectives," *Foundations and Trends in Databases Series*, 2016, http://data-people.cs.illinois.edu/crowd-book.html.

[3] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *Intelligent Systems, IEEE*, 2009.

[4] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM*, 2012.

[5] A. G. Parameswaran, et al., "Crowdscreen: algorithms for filtering data with humans," in *SIGMOD*, 2012.

[6] A. G. Parameswaran, S. Boyd, et al., "Optimal crowd-powered rating and filtering algorithms," in *VLDB*, 2014.

[7] A. Marcus, E. Wu, D. R. Karger, S. Madden, and R. C. Miller, "Human-powered Sorts and Joins," *PVLDB*, 2011.

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

[9] V. Polychronopoulos, L. de Alfaro, et al., "Human-powered top-k lists," in *WebDB*, 2013.

[10] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "CrowdER: Crowdsourcing Entity Resolution," *PVLDB*, 2012.

[11] S. Guo, A. Parameswaran, et al., "So who won? dynamic max discovery with the crowd," in *SIGMOD*, 2012.

[12] D. E. Difallah, et al., "The dynamics of micro-task crowdsourcing: The case of amazon mturk," in *WWW*, 2015.

[13] E. Pavlick, J. Bos, M. Nissim, C. Beller, B. Van, and D. C. Callison-burch, "Adding semantics to data-driven paraphrasing," in *In ACL*, Citeseer, 2015.

[14] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *EMNLP*, Association for Computational Linguistics, 2015.

[15] A. Vedaldi, S. Mahendran, et al., "Understanding objects in detail with fine-grained attributes," in *CVPR*, 2014.

[16] S. Bell, K. Bala, and N. Snavely, "Intrinsic images in the wild," *ACM Transactions on Graphics (TOG)*, 2014.

[17] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *CVPR, 2012*.

[18] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2014.

[19] A. D. Sarma, A. Jain, A. Nandi, A. Parameswaran, and J. Widom, "Surpassing humans and computers with jellybean: Crowd-vision-hybrid counting algorithms," in *HCOMP*, 2015.

[20] B. Plummer, L. Wang, C. Cervantes, J. Caicedo, J. Hockenmaier, and S. Lazebnik, "Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models," *CoRR*, vol. abs/1505.04870, 2015.

[21] S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed transformation," *OPTICAL ENGINEERING-NEW YORK-MARCEL DEKKER INCORPORATED-*, vol. 34, pp. 433–433, 1992.

[22] T. W. Nattkemper, H. Wersing, W. Schubert, and H. Ritter, "A neural network architecture for automatic segmentation of fluorescence micrographs," *Neurocomputing*, vol. 48, no. 1, pp. 357–367, 2002.

[23] M. Dyer and A. Frieze, "On the complexity of partitioning graphs into connected subgraphs," *Discrete Applied Mathematics*, vol. 10, no. 2, pp. 139–153, 1985.

[24] E. G. Coffman Jr, M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: a survey," in *Approximation algorithms for NP-hard problems*, pp. 46–93, PWS Publishing Co., 1996.

[25] R. Gomes, P. Welinder, A. Krause, and P. Perona, "Crowdclustering," in *NIPS*, pp. 558–566, 2011.

[26] J. Yi, R. Jin, A. K. Jain, and S. Jain, "Crowdclustering with sparse pairwise labels: A matrix completion approach," in *In AAAI Workshop on Human Computation*, 2012.

[27] A. G. Parameswaran, A. D. Sarma, H. Garcia-Molina, N. Polyzotis, and J. Widom, "Human-assisted graph search: it's okay to ask questions," *PVLDB*, vol. 4, no. 5, pp. 267–278, 2011.

[28] B. Trushkowsky, T. Kraska, M. J. Franklin, and P. Sarkar, "Crowdsourced enumeration queries," in *ICDE*, 2013.

[29] T. Rekatsinas, A. Deshpande, and A. Parameswaran, "CrowdGather: Entity Extraction over Structured Domains," *ArXiv e-prints*, Feb. 2015.

[30] A. Parameswaran, M. H. Teh, H. Garcia-Molina, and J. Widom, "DataSift: An Expressive and Accurate Crowd-Powered Search Toolkit," in *HCOMP*, 2013.

[31] A. G. Parameswaran, M. H. Teh, H. Garcia-Molina, and J. Widom, "Datasift: a crowd-powered search toolkit," in *International Conference on Management of Data, SIGMOD*, 2014.

[32] H. Zhuang, A. G. Parameswaran, D. Roth, and J. Han, "Debiasing crowdsourced batches," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

[33] D. Yu and L. Deng, *Automatic Speech Recognition*. Springer, 2012.

[34] W. Lasecki, C. Miller, A. Sadilek, A. Abumoussa, D. Borrello, R. Kushalnagar, and J. Bigham, "Real-time captioning by groups of non-experts," in *UIST*, 2012.

[35] J. D. Williams, I. D. Melamed, T. Alonso, B. Hollister, and J. Wilpon, "Crowd-sourcing for difficult transcription of speech," in *Automatic Speech Recognition and Understanding (ASRU)*, 2011.

[36] R. Van Dalen, K. Knill, P. Tsiakoulis, and M. Gales, "Improving multiple-crowd-sourced transcriptions using a speech recogniser," in *ICASSP*, 2015.

[37] O. Russakovsky, L.-J. Li, and L. Fei-Fei, "Best of both worlds: human-machine collaboration for object annotation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2121–2131, 2015.

[38] I. Lotosh, T. Milo, and S. Novgorodov, "Crowdplanr: Planning made easy with crowd," in *ICDE*, 2013.

[39] Y. Amsterdamer, Y. Grossman, T. Milo, and P. Senellart, "Crowd mining," in *SIGMOD Conference*, 2013.

[40] G. Demartini, B. Trushkowsky, T. Kraska, and M. J. Franklin, "Crowdq: Crowdsourced query understanding," in *CIDR*, 2013.

[41] Y. Gao and A. G. Parameswaran, "Finish them!: Pricing algorithms for human computation," *PVLDB*, 2014.

[42] P. Venetis and H. Garcia-Molina, "Dynamic max algorithms in crowdsourcing environments," technical report, Stanford University, August 2012.

[43] P. Venetis, H. Garcia-Molina, K. Huang, and N. Polyzotis, "Max algorithms in crowdsourcing environments," in *Proceedings of the 21st World Wide Web Conference*, 2012.

[44] S. B. Davidson, S. Khanna, T. Milo, and S. Roy, "Using the crowd for top-k and group-by queries," in *ICDT*, 2013.

[45] D. Deng, C. Shahabi, U. Demiryurek, and L. Zhu, "Task selection in spatial crowdsourcing from worker's perspective," *GeoInformatica*, vol. 20, no. 3, pp. 529–568, 2016.

[46] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Trans. Spatial Algorithms and Systems*, vol. 1, no. 1, p. 2, 2015.

[47] G. Demartini, D. E. Difallah, and P. Cudré-Mauroux, "ZenCrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking," in *WWW*, 2012.

[48] X. Chu, J. Morcos, I. F. Ilyas, M. Ouzzani, P. Papotti, N. Tang, and Y. Ye, "Katara: A data cleaning system powered by knowledge bases and crowdsourcing," in *SIGMOD*, 2015.

[49] J. Wang, G. Li, T. Kraska, M. J. Franklin, and J. Feng, "Leveraging transitive relations for crowdsourced joins," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 229–240, ACM, 2013.

[50] C. Gokhale, S. Das, A. Doan, J. F. Naughton, N. Rampalli, J. Shavlik, and X. Zhu, "Corleone: Hands-Off Crowdsourcing for Entity Matching," *SIGMOD 2014*, Mar. 2014.

[51] S. Wang, X. Xiao, and C.-H. Lee, "Crowd-based deduplication: An adaptive approach," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pp. 1263–1277, ACM, 2015.

[52] K. Bellare, S. Iyengar, A. Parameswaran, and V. Rastogi, "Active sampling for entity matching with guarantees," in *ACM Transactions on Knowledge Discovery from Databases (TKDD)*, 2013.

[53] K. Bellare, S. Iyengar, A. G. Parameswaran, and V. Rastogi, "Active sampling for entity matching," in *KDD*, 2012.

[54] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, "Crowddb: answering queries with crowdsourcing," in *SIGMOD Conference*, pp. 61–72, 2011.

[55] A. Marcus, et al., "Crowdsourced databases: Query processing with people," in *CIDR*, 2011.

[56] A. G. Parameswaran, H. Park, H. Garcia-Molina, N. Polyzotis, and J. Widom, "Deco: declarative crowdsourcing," in *CIKM*, pp. 1203–1212, 2012.

[57] H. Park, A. Parameswaran, and J. Widom, "Query processing over crowdsourced data," technical report, Stanford University, September 2012.

[58] A. Bozzon, M. Brambilla, and S. Ceri, "Answering search queries with crowdsearcher," in *WWW*, 2012.

[59] D. Deutch, et al., "Using markov chain monte carlo to play trivia," in *ICDE*, 2011.

[60] D. R. Karger, S. Oh, and D. Shah, "Budget-optimal task allocation for reliable crowdsourcing systems," in *CoRR*, vol. abs/1110.3564, 2011.

[61] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. R. Movellan, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise," in *NIPS*, pp. 2035–2043, 2009.

[62] N. Dalvi, A. Dasgupta, R. Kumar, and V. Rastogi, "Aggregating crowdsourced binary ratings," in *WWW*, 2013.

[63] Y. Zhang, et al., "Spectral methods meet em: A provably optimal algorithm for crowdsourcing", *arXiv preprint*, 2014.

[64] A. Ramesh, A. Parameswaran, H. Garcia-Molina, and N. Polyzotis, "Identifying reliable workers swiftly," technical report, Stanford University, September 2012.

[65] M. Joglekar, H. Garcia-Molina, and A. Parameswaran, "Evaluating the crowd with confidence," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 686–694, ACM, 2013.

[66] M. Joglekar, H. Garcia-Molina, and A. Parameswaran, "Comprehensive and reliable crowd assessment algorithms," in *ICDE*, 2015.

[67] A. Das Sarma, A. Parameswaran, and J. Widom, "Towards globally optimal crowdsourcing quality management: The uniform worker setting," in *SIGMOD*, 2016.

[68] C. H. Lin, Mausam, and D. S. Weld, "Crowdsourcing control: Moving beyond multiple choice," in *UAI*, 2012.

[69] E. Kamar, et al., "Combining human and machine intelligence in large-scale crowdsourcing," in *AAMAS*, 2012.