We propose a novel three-stage FIND-RESOLVE-LABEL workflow for crowdsourced annotation in order to reduce ambiguity of task instructions and thereby improve annotation accuracy. Stage 1 (FIND) asks the crowd to find examples whose correct label seems ambiguous given task instructions. Workers are also asked to provide a short tag which describes the ambiguous concept embodied by the specific instance found, and we compare collaboration vs. non-collaboration designs for this stage. Next, in Stage 2 (RESOLVE), the Requester selects one or more of these ambiguous examples to label. These are then automatically injected back into task instructions in order to improve clarity. Finally, in Stage 3 (LABEL), workers perform the actual annotation using the revised guidelines with clarifying examples. We compare three designs for using these examples: specific examples, conceptual tags, or both. We report image labeling experiments on six task designs conducted on Amazon’s Mechanical Turk. Results show that improved annotation accuracy and other insights regarding effective design for crowdsourced annotation tasks. To enable reproducibility and follow-on studies by others, we share all of our data online.

KEYWORDS

crowdsourcing, annotation, design, human-computer interaction

1 INTRODUCTION

While crowdsourcing now enables labeled data to be obtained more quickly, cheaply, and easily than ever before [2, 42, 43], ensuring data quality remains something of an art, challenge, and perpetual risk. Consider a typical workflow for annotating data on Amazon Mechanical Turk (MTurk): a Requester designs an annotation task, asks multiple workers to complete it, and then aggregates their labels to induce consensus labels. Because the annotation work itself is largely opaque, with only submitted labels being observable, the Requester has little insight into what if any problems workers encounter during annotation. While aggregation methods [20, 41, 47] can be employed to eliminate random errors and some amount of spam, if many of the workers are actually confused by incomplete, unclear, or ambiguous task instructions, how much improvement can one realistically expect from aggregation after-the-fact?

In contrast, consider a more traditional annotation workflow involving trusted annotators, such as practiced by the Linguistic Data Consortium (LDC) [19]. Once preliminary annotation guidelines are developed, an iterative process ensues in which: 1) a subset of data is labeled based on current guidelines; 2) annotators review corner cases and disagreements, review relevant guidelines, and reach consensus on appropriate resolutions; 3) annotation guidelines are updated; and 4) the process repeats. In comparison to the simple crowdsourcing workflow above, this traditional workflow iteratively debugs and refines task guidelines for clarity and completeness in order to deliver higher quality annotations. However, it comes at the cost of more overhead, with a heavier process involving open-ended interactions with trusted annotators. Could we somehow combine these for the best of both worlds?

In this work, we propose a novel three-stage FIND-RESOLVE-LABEL design pattern for crowdsourced annotation which strikes a middle-ground between the efficient crowdsourcing workflow on one hand and the high quality LDC-style workflow on the other. To improve data quality, we engage workers in helping to debug and clarify annotation guidelines, yet we do so via a light-weight, highly-structured task design suitable for efficient crowd work. Overall, we design the annotation process as a partnership between Requester and workers in which we each party has different strengths and responsibilities. To ensure data quality while preserving efficiency, we seek to maximize the relative strengths of each party.

Figure 1 depicts our overall workflow. In Stage 1 (FIND), workers are shown initial guidelines for an annotation task and asked to search for data instances which appear ambiguous given the guidelines. For each instance workers find, they are also asked to provide a short tag which describes the concept embodied by the specific instance which is ambiguous given the guidelines. Next, in Stage 2 (RESOLVE), the Requester selects one or more of the ambiguous instances to label as exemplars. Those instances and their tags are then automatically injected back into the annotation guidelines in order to improve clarity. Finally, in Stage 3 (LABEL), workers perform the actual annotation using the revised guidelines with clarifying examples. The Requester can run the LABEL stage on a sample of data, assess label quality, and then decide how to proceed. If quality is sufficient, remaining data can simply be labeled according to the guidelines. Otherwise, Stages 1 and 2 can be iterated in order to further refine the guidelines.

To evaluate our Three-stage task design, we construct 6 different image labeling tasks. These tasks have different levels of difficulty and intuitiveness. We construct a test dataset which contains different ambiguous and unambiguous concepts. Starting from simple and possibly ambiguous task instructions, we then improve instructions via our three-stage workflow. Given expert (gold) labels for our dataset for each of the 6 tasks, we can evaluate how well revised instructions compare to original instructions by measuring the accuracy of the labels obtained from the crowd.

Contributions. We show that the crowd can deliver informative ambiguous examples which can be used to further clarify task instructions, and that these examples can be utilized to improve
Our Three-Stage FIND-RESOLVE-LABEL workflow is shown above. Stage 1 (FIND) asks the crowd to find examples whose correct label seems ambiguous given task instructions. In Stage 2 (RESOLVE), the Requester selects one or more of these ambiguous examples to label. These are then automatically injected back into task instructions in order to improve clarity. Finally, in Stage 3 (LABEL), workers perform the actual annotation using the revised guidelines with clarifying examples. If Stage 3 labeling quality is insufficient, we can return to Stage 1 to find more ambiguous examples to further clarify instructions.

2 MOTIVATION AND BACKGROUND

Consider the task of labeling images for object detection. For example, on MTurk one might post a task such as, “Is there a dog in this image?” Such a task appears to be quite simple, but is it? For example, is a wolf a dog? What about more exotic and unusual wild breeds of dogs? Does the dog need to be a real animal or merely a depiction of one? What about a museum model of an ancient but extinct dog breed, or a realistic wax sculpture? What if the dog is only partially visible in the image? Ultimately, what is it that the Requester really wants? For example, a Requester interested in anything and everything dog-related might have very liberal inclusion criteria. On the other hand, annotators bring with them their own diversity of implicit biases which the Requester may not be aware of or understand. The Requester might be ignorant themselves of the domain (e.g., is a wolf a type of dog?) or have not fully-defined what they are looking for. For example, in information retrieval, users’ own conception and understanding of what they are looking for often evolves during the process of search and browsing [12]. We describe our own experiences with this in Section 5.1.2.

2.1 Helping Requesters Succeed

Best practices. A variety of tutorials, surveys, introductions, and research papers offer how-to advice for successful microtask crowdsourcing with platforms such as MTurk [14, 22, 28, 33]. For example, it is often recommended that Requesters invest time browsing and labeling some data themselves before launching a task in order to better define and debug it [2]. Studies have compared alternative task designs to suggest best practices [18, 25, 36, 46].

Templates and Assisted Design. Rather than start task design from scratch, MTurk now offers templates and has suggested Requesters share successful templates for others’ use [10]. Similarly, famous research on software design patterns [16] has inspired ideas for similar crowdsourcing design patterns which could be reused across different data collection tasks. For example, FIND-FIX-VERIFY [7] is a well-known example that partially inspired our work. Other researchers have suggested improved tool support for workflow design [26] or engaging the crowd itself in task design or decomposition [27, 30].
2.2 Understanding disagreement

Random Noise vs. Bias. Since annotators are human, even trusted annotators will naturally make mistakes from time-to-time. Fortunately, random error is exactly the kind of disagreement that aggregation [20, 41, 47] can easily resolve; assuming such mistakes are relatively infrequent and independent, workers will rarely err at the same, and so something as simple as majority voting can resolve disagreement. On the other hand, if workers have individual biases, they will make consistent errors; e.g., a teenager vs. a protective mother might have liberal vs. conservative biases in rating movies [21]. In this case, it is useful to detect such consistent biases and recalibrate worker responses to undo such bias. Aggregation can also work provided that workers do not share the same biases. However, when workers do share systematic biases, e.g., the independence assumption underlying aggregation is violated, and so aggregation can amplify bias rather than resolve it. Consequently, it is important that annotation guidelines identify cases in which annotator biases conflict with desired labels and particularly establish clear expectations for how such cases should be handled.

Objective vs. Subjective tasks. In a fully-objective tasks, we assume each question has a single correct answer, and any disagreement with the gold standard reflects error. Label aggregation methods largely operate in this space. Tasks in which each question has a single correct answer. At the other extreme, purely-subjective (i.e., opinion) tasks permit a wide range of valid responses with little expectation of agreement between individuals (e.g., asking one’s favorite color or food). Between these simple extremes, however, lies a wide, interesting, and important space of partially-subjective tasks in which answers are only partially-constrained [35, 40, 44]. For example, consider rating item quality: while agreement tends be high for items having extremely good or bad properties, instances with more middling properties naturally elicit a wider variance in opinion. In general, because subjectivity permits a valid diversity of responses, it can be difficult to detect if a judge does not undertake the task in good faith, complicating quality assurance.

Difficulty vs. Ambiguity. Some annotation tasks are more complex than others, just as some instances within each task are more difficult to label than other instances. A common concern with crowdsourcing is whether inexpert workers have sufficient expertise to successfully undertake a given annotation tasks. Intuitively, more guidance is likely necessary with more skilled tasks and less expert workers. Alternatively, if we use sufficiently expert annotators, we assume difficult cases can be handled. With ambiguity, on the other hand, it would be unclear even to an expert what to do. Ambiguity is an interaction between data instances and annotation guidelines; effectively, an ambiguous instance is a corner-case wrt. guidelines. Aggregation can helpfully identify the majority interpretation, but that interpretation may or may not be what is actually desired (e.g., recall systematic bias discussion above).

Static vs. Dynamic Disagreement. As annotators undertake a task, their understanding of work evolves as they develop familiarity with both the data and the guidelines. In fact, prior work has shown that annotators interpret and implement task guidelines in different ways as annotation progresses [24, 39]. Consequently, different sorts of disagreement can occur at different stages of annotation. Temporal-aware aggregation can partially ameliorate this [23], as can implementing data collection processes to train, “burn-in”, or calibrate annotators, controlling and/or accelerating their transition them from an initial learning state into a steady state. For example, we emphasize identifying key boundary-cases and expected labels for them.

2.3 Crowdsourcing Beyond Data Labeling

While data annotation is the canonical use of crowdsourcing in regard to training and evaluating machine learning models, human intelligence can be tapped in a much wider and more creative set of ways than we often consider. For example, the crowd might suggest useful features for a machine learning classifier [11].

One of the oldest crowdsourcing design patterns is utilizing the scale of the crowd to help search or filter a large space of possibilities2. This pattern has been used in many crowdsourcing task designs since, such as in Bernstein et al. [7]’s FIND-FIX-VERIFY, or physical search, such as in DARPA’s Network Challenge [37] or human flesh search [45]. Our work was especially inspired by Attenberg et al. [5], who ask the crowd to search for examples for which a machine learning model is confident but wrong. Our calling upon the crowd to search for ambiguous examples given task guidelines further explores the potential of this same crowd design pattern. Rather than wait for ambiguous examples to be encountered by chance during annotation, we instead rapidly identify corner-cases by explicitly searching for them. We offload to the crowd the tedious work of searching data for ambiguous cases, and who better to identify potentially ambiguous examples than the same workforce will be asked to perform the actual annotation? At the same time we reduce Requester work, limiting their effort to labeling corner-cases, we also show the crowd that understand the importance of task clarity and that we value the importance of creating clear tasks for workers (e.g., see Figure 2).

3 WORKFLOW DESIGN

We propose a three-stage FIND-RESOLVE-LABEL task in this work. An illustration of the workflow is shown in Figure 1. In Stage 1 (FIND), the workers are asked to collect ambiguous examples and concept tags given task instructions (Section 3.1). Next, in Stage 2 (RESOLVE), the Requester selects one or more of the ambiguous examples found by the crowd to label. These labeled examples are then automatically injected back into task instructions in order to further improve clarity (Section 3.2). Finally, in Stage 3 (LABEL), workers perform the actual annotation using the revised guidelines with clarifying examples (Section 3.3).

The Requester can run the final LABEL stage on a sample of data, assess label quality, and then decide how to proceed. If quality is sufficient, remaining data can simply be labeled according to the

2 e.g., https://www.allthingsdistributed.com/2007/02/help_find_jim_gray.html
In the Stage 1 (FIND) task, workers are asked to search for examples they think would be ambiguous given task instructions. In this case, “Is there a dog in this image?” In collaboration conditions (Section 3.1.1), workers will see additional ambiguous examples found by past workers.

Figure 2 shows the main task interface for Stage 1 (FIND). The interface shows the annotation task to the workers (e.g., “Is there a dog in this image?”) and asks them “Can you find ambiguous examples for this task?” Pilot experiments found that workers had difficulty understanding the task based on only this prompt, so we made a further assumption that the Requester would provide a single example of such an ambiguity in order to help clarify the FIND task for the workers. For example, for the dog annotation task, we show an image of a Toy Dog. Workers are then directed to use Google Image Search to find these ambiguous examples. Once an ambiguous image is uploaded, another page (not shown) asks workers to provide a short tag which explains the ambiguity.

3.1 Stage 1: Finding Ambiguous Examples

In Stage 1 (FIND), workers are asked to collect ambiguous examples and concept tags given task instructions. The tag serves multiple purposes. Firstly, it acts like a crowd rationale [34] in providing quality control, requiring workers to justify their answers providing a form of transparency to help Requesters better understand worker intent. Secondly, the tag provides a conceptual explanation of the ambiguity which can then be re-injected into annotation guidelines to help explain corner-cases to future workers.

Figure 2 shows the main task interface for Stage 1 (FIND). The interface shows the annotation task to the workers (e.g., “Is there a dog in this image?”) and asks them “Can you find ambiguous examples for this task?” Pilot experiments found that workers had difficulty understanding the task based on only this prompt, so we made a further assumption that the Requester would provide a single example of such an ambiguity in order to help clarify the FIND task for the workers. For example, for the dog annotation task, we show an image of a Toy Dog. Workers are then directed to use Google Image Search to find these ambiguous examples. Once an ambiguous image is uploaded, another page (not shown) asks workers to provide a short tag which explains the ambiguity.

3.1.1 Exploring Collaboration. While crowd work has traditionally been performed independently, a variety of work has explored collaboration mechanisms by which workers might usefully help each other complete a task more effectively [8, 13, 31]. To investigate the potential value of worker collaboration in finding higher quality ambiguities, we explore a light-weight design in which workers do not directly interact but are shown examples found by past workers (in addition to the example provided by the Requester). Operating in sequential, snowball fashion, worker2 would see worker1’s example, while worker3 would see examples found by worker1 and worker2, etc. In all variants below, the worker is shown one or more one ambiguous example(s) with associated conceptual tag(s) and asked to find another, different example of ambiguity, along with a conceptual tag for the example found.

1. Baseline: No Collaboration. Each worker sees the task design and one ambiguous example with ambiguity tag provided by the Requester. Workers must find ambiguities independently.

2. Unfiltered Collaboration. Workers see all ambiguous examples (and their tags) found by past workers. There is no quality control, so workers may see incorrect and/or duplicated examples.

3. Filtered Collaboration. Given a set of ambiguous examples found by the crowd, the Requester chooses the best ones and only these are shown to subsequent Stage 1 workers. Note that this design requires Requester involvement. In our design, the Requester intervenes and chooses the best examples after every batch of 5 ambiguous examples are found. In other words, after each set of 5 workers have provided their examples, the Requester chooses the best examples, which are then incorporated back into the task page. We adopt essentially the same Requester UI for this as in Stage 2 (see Figure 3), albeit the Requester does not label examples but merely selects which examples to keep via simple mouse click.

3.2 Stage 2: Resolving Ambiguous Examples

After collecting the ambiguous examples from Stage 1, the Requester then selects one or more of these ambiguous examples to label. These are labeled examples are then automatically injected back into task instructions in order to improve clarity.

Figure 3: For Stage 2 (RESOLVE), our interface design lets a Requester easily select and label images. Each mouse click on an example toggles between unselected, selected positive, and selected negative states.
A simple, easy, and fast Requester interface design for Stage 2 is shown in Figure 3. The Requester need only click on examples to keep, with the same mouse clicks also providing example labels.

### 3.3 Stage 3: Labeling with Clarifying Examples

Best practices suggest that along with task instructions, Requesters should include a few examples and their correct annotations [46]. However, randomly selected examples may not be as helpful to workers as specifically targeting ambiguous corner-cases to show. We hypothesize that showing the workers key ambiguous examples will help them learn the task better.

Following the RESOLVE stage, we automatically append to task instructions the key examples selected by the Requester, along with their labels. Choices made by the Requester in Stage 2 (shown in Figure 3), lead to automatic generation of the partial interface shown in Figure 4. Positive examples are shown first (“you should select concepts like these”), followed by negative examples below that (“and NOT select examples like these”). Note that no task-specific instructions or layout are required; we simply inject the selected examples into the template as shown.

You should select concepts like these:

Dog

and NOT select concepts like these:

Cartoon Dog  Wolf  Statue  Robot Dog

Figure 4: For Stage 3 (LABEL), we combine the ambiguous instances and/or tags collected in Stage 1 (FIND) with the Requester labels from Stage 2 (RESOLVE) and automatically inject the labeled examples back into task instructions.

Presenting Examples to Workers. To assess the relative utility of collecting ambiguous examples vs. conceptual tags for ambiguous concepts, we evaluate three different designs for how to present these examples to workers: 1) only ambiguous examples are shown (AMB, no conceptual tags); 2) only tags for ambiguous concepts are shown (TAG, no example images); and 3) both ambiguous examples and their conceptual tags are shown (AMB+TAG).

### 4 EXPERIMENTAL SETUP

Experiments are conducted for six different annotation tasks related to binary labeling tasks in regard to the presence or absence of dog-related categories. As we shall see, even such a seemingly simple domain reveals a variety of subtle nuances in practice.

**Dataset.** All experiments utilize the same set of 40 images we found via a combination of 1) our own Google searches; and 2) images found by workers in Stage 1 which they deemed ambiguous given the task instructions “Is there a dog in this image?” The total image set was designed to encompass both easy, unambiguous images and a range of diverse, difficult and/or ambiguous images. Similar in spirit to Kulesza et al. [29], we identified a set of underlying, dog-related categories, identified via multiple passes over the data. We organized images into 11 categories: dogs, small dog breeds, similar animals, cartoons, stuffed toys, robots, statues, dog-related objects (e.g., dog-shaped cloud), miscellaneous (e.g., hot dog, word “dog”), different animals (difficult to confuse for dogs) and planes (the easiest non-dog category workers should never confuse). We assigned each image to exactly one category.

**Gold Labels.** For each of our six, binary annotation tasks, we partitioned these categories into positive vs. negative categories. We then measured worker accuracy in correctly labeling images according to positive and negative categories for each task.

**Negative Qualifications.** We utilize “negative” qualifications [4] on MTurk to prevent any worker from performing more than one task for us, thus avoiding any confounding learning effects.

**The Requester.** In this work, we act as the Requester. This includes specifying task instructions and intents (see next Section), and in performing Stage 2 work in selecting which examples to use for Stage 3, using gold labels as specified above.

### 4.1 Annotation Tasks

When a user searches the Web with a query such as “apple”, are they looking for information about the fruit, the company, or something else entirely? Despite the paucity of detail provided by a typical terse query, search result accuracy is assessed based on how well results match the user’s underlying intent. Further complicating matters, the user’s notion of what is relevant may not be well-defined in advance of their search, but rather evolve as they browse search results and learn more about the given topic. How people learn as they search remains an open question [17, 38].

Similarly, Requesters expect workers to understand the annotation “intent” underlying the explicit instructions they give, and they evaluate worker accuracy wrt. to understanding that intent, even if instructions are incomplete, unclear, or ambiguous. To capture this in our evaluation, we designed 3 different annotation tasks where, in each case, the instructions are ambiguous in terms of having at least two possible interpretations. As such, adding examples to instructions can help clarify Requester intent to workers.

Anecdotally, a major crowdsourcing company providing an image moderation service faced a challenge in which different customers had slight variants in rule sets (e.g., a picture of a man without a shirt was considered acceptable by one customer while another customer deemed such images inappropriate unless the man was outdoors). Consequently, we know in practice that workers must sometimes memorize arbitrary variants of classification rules for different Requesters, and these rules may be more or less intuitive to each worker given their personal world-views.

Similarly, we design our tasks in this work such that different intents are more or less intuitive, letting us assess the effectiveness of our design under intuitive intents vs. more esoteric intents which may force workers to learn seemingly arbitrary rule patterns which may conflict with their initial assumptions about Requester intent.
We expect it will be more difficult to teach workers to perform less intuitive labeling tasks.

For each task and intent below, we list positive categories; all other categories are assumed to be negative for that intent.

4.1.1 Task 1: Is there a dog in this image?
Intent a (more intuitive): dogs and small dog breeds
Intent b (less intuitive): dogs, small dog breeds, similar animals.
Rationale: The Requester wants to train a deep learning model for avoiding animals and believes the model may also benefit from seeing images of wolves and foxes in the training data labeled.

4.1.2 Task 2: Is there a fake dog in this image?
Intent a: similar animals. Rationale: The Requester is looking for animals often confused with dogs.

4.1.3 Task 3: Is there a toy dog in this image?
Intent a (less intuitive): small dog breeds. Rationale: As we ourselves learned in the course of this work (Section 5.1.2), small dogs, such as Chihuahua or Yorkshire Terrier, are collectively known as "toy dog" breeds. Fewer people may be familiar with this.
Intent b (more intuitive): stuffed toys, robots. Rationale: The Requester is looking for children’s toys, e.g., to train a model for an e-commerce site.

4.2 Evaluating Stage 1 (FIND)
For Stage 1, we collect worker submissions in providing ambiguous images (and conceptual tags) for Task 1: "Is there a dog in this image?". We evaluate these based on three criteria of increasing strictness: 1) correctness; 2) uniqueness; and 3) usefulness.

Correctness captures our assessment of whether the worker appeared to have understood the task correctly and submitted a plausible example of ambiguity for the given task. Any incorrect examples are excluded from consideration for uniqueness or usefulness.

Uniqueness measures our subjective assessment of how many distinct types of ambiguity workers found across correct examples. For example, we deemed “Stuffed Dog” and “Toy Dog” sufficiently close as to represent the same concept.

Usefulness assesses which of the unique ambiguous concepts found are likely to be useful in annotation. For example, while an image of a hot dog is valid and unique, it is unlikely that many annotators would find it ambiguous in practice.

In all three collaboration conditions: no collaboration, unfiltered collaboration, and filtered collaboration (Section 3.1.1), 15 ambiguous examples are collected and evaluated for the above criteria.

4.3 Evaluating Stage 3 (LABEL)
We design our Stage 3 MTurk human intelligence task (HIT) as follows. A worker is shown 10 images and is asked to select all images that have positive labels. We pay $0.06 per HIT, and use 9 assignments (i.e., workers) per HIT.

As discussed in Section 3.3, we consider three different designs for how to present ambiguous examples and concepts to workers: 1) only ambiguous examples are shown (AMB, no conceptual tags); 2) only tags for ambiguous concepts are shown (TAG, no example images); and 3) both ambiguous examples and their conceptual tags are shown (AMB+TAG). In addition, we also consider two baselines. The first baseline (B0) assess the utility of including any examples at all. B0 includes only the task instructions; no examples are shown. The second baseline (B1) assesses the value of intelligently selecting which examples to show. B1 selects random examples to show, whereas AMB shows Stage 2 selected ambiguous examples.

5 RESULTS
5.1 Can Workers Find Ambiguous Concepts?
5.1.1 Challenges. Our initial pilot runs of Stage 1 (FIND) encountered two main issues: 1) duplicate concepts; and 2) workers misunderstanding the FIND task. In regard to duplicate concepts, some easy-to-find and closely-related concepts were naturally repeated multiple times. However, also recall from Section 3.1 that we assume Requesters will provide a single ambiguous example in order to clarify the FIND task to workers. Combining (1) and (2), some workers misunderstood this to mean they should provide more examples of the same ambiguous concept. As for more general worker misunderstanding (2), some workers appeared to have searched Google Image Search for "ambiguous image", submitting generally ambiguous images, rather than images that were ambiguous given the specific task instructions, as intended.

5.1.2 Qualitative Results. A best practice that is not always heeded is for Requesters to invest time browsing and labeling some data themselves before launching a task in order to better define and debug their own task [2]. One of the reasons Requesters may skip this, or not invest sufficient time, is the scale and tedium of sifting through data. Another challenge of this scale is that it may be hard for Requesters to discover rare concepts which violate their prior expectations or trained models [5].

In running our experiments, we found that workers helped us to learn about our own task, data, ignorance, and biases. For example, in pilot experiments, one Stage 1 worker returned an image of a Chihuahua (small breed of dog) as being ambiguous as a "toy dog". At first we did not understand the source of the confusion. Later, however, we learned small dog breeds are called "toy dogs". Prior to that, our conception of the concept was limited to children’s toys. Hence, this inspired our evaluation tasks - “Is there a Toy Dog in this image?” with two different interpretations of Toy Dog in Tasks 3a & 3b (Section 4.1).

Our favorite example, however, was when a worker submitted a picture of a man as an ambiguous example (see Figure 5). As above, we initially jumped to the conclusion that the worker’s response was spam, only to later discover our new favorite reality show celebrity, “Dog the Bounty Hunter”.

These experiences served as an excellent example of how we Requesters may easily jump to a conclusion about worker spam due to our own ignorance or inability to recognize the diversity of valid perspectives on seemingly objective tasks [40, 44]. It should not be surprising that Requesters and workers may have different social and cultural backgrounds, making it difficult for Requesters to imagine all possible sources for confusion in their task instructions.

https://en.wikipedia.org/wiki/Toy_dog

http://www.dogthebountyhunter.com
We now report results of using ambiguous examples found in Stage 1 (FIND) for the task “Is there a dog in this image?”. We capitalize tags here for presentation but use raw worker tags without modification in our evaluation.

Who better to identify potentially ambiguous examples than the workers in Stage 1 (FIND) for the task “Is there a dog in this image”? We compare three different designs for automatically incorporating the images and tags collected in the previous step.

Table 1: Evaluating Collaboration for Stage 1 (FIND).

<table>
<thead>
<tr>
<th></th>
<th>Correct</th>
<th>Unique</th>
<th>Useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Collaboration</td>
<td>60.0</td>
<td>26.7</td>
<td>26.7</td>
</tr>
<tr>
<td>Unfiltered Collaboration</td>
<td>93.0</td>
<td>40.0</td>
<td>33.3</td>
</tr>
<tr>
<td>Filtered Collaboration</td>
<td>60.0</td>
<td>20.0</td>
<td>13.3</td>
</tr>
</tbody>
</table>

Table 2 summarizes our main findings.

5.2.1 Using Examples to Teach Annotation Guidelines. We know that providing examples to the workers helps them perform the task better [46]. Comparing designs B0 and B1 in Table 2, we clearly see that providing examples (B1) almost always produces more accurate labeling than a design that provides no examples (B0). In addition to this, the design IMG performs better than B1, this shows that the kind of examples that are provided is important. Showing the workers ambiguous examples is clearly superior to showing them randomly chosen examples. This supports our hypothesis: ambiguous examples appear to delineate labeling boundaries for the task better than random examples.

5.2.2 Instances vs. Concepts. Best practices suggest that Requesters provide examples when designing their task [46]. We include this design in our evaluation as B1. An alternate design is to show concepts as examples instead of specific instances; this is our design TAG, shown in Table 2. For example, for the task “Is there a Dog in this image?”, instead of showing a dog statue image, we could simply provide the example concept “Inanimate Objects” shown concept tags along with an example image for that concept well. This might be overcome by better selecting a more representing example for a concept, or showing more examples for each concept. We leave such questions for future work.

5.3 Analyzing Sources of Worker Errors

5.3.1 Difficult vs. Subjective Questions. Accuracy for categories “Similar Animal” and “Cartoon” for Task 1b (Section 4.1) is shown in Table 3. We see that some concepts appear more difficult, such as correctly labeling a wolf or a fox. Annotations appear to need some world knowledge or training of differences species in order to correctly distinguish such examples vs. dogs. The results show that such concepts are more difficult to teach to the crowd; even though the accuracy improves, the improvement is less than we see with other concepts. In contrast, for Cartoon Dog (an example of a subjective question), adding this category to the illustrative examples greatly reduces the ambiguity for annotators. Other concepts like “Robot” and “Statue” also show large improvements in accuracy.
Table 2: Comparison of Task Designs across the Six Tasks (Section 4.1 based on Annotation Accuracy.

<table>
<thead>
<tr>
<th>Design</th>
<th>Task 1a</th>
<th>Task 1b</th>
<th>Task 2a</th>
<th>Task 2b</th>
<th>Task 3a</th>
<th>Task 3b</th>
</tr>
</thead>
<tbody>
<tr>
<td>B0</td>
<td>75.6</td>
<td>70.1</td>
<td>47.8</td>
<td>78.7</td>
<td>69.1</td>
<td>92.9</td>
</tr>
<tr>
<td>B1</td>
<td>83.0</td>
<td>66.4</td>
<td>59.0</td>
<td>85.5</td>
<td>78.7</td>
<td>96.0</td>
</tr>
<tr>
<td>IMG</td>
<td>88.0</td>
<td>89.5</td>
<td>68.5</td>
<td>85.8</td>
<td>89.2</td>
<td>93.5</td>
</tr>
<tr>
<td>TAG</td>
<td>91.0</td>
<td>91.0</td>
<td>79.0</td>
<td>87.0</td>
<td>91.0</td>
<td>96.9</td>
</tr>
<tr>
<td>IMG+TAG</td>
<td>91.4</td>
<td>87.0</td>
<td>81.8</td>
<td>86.4</td>
<td>88.3</td>
<td>96.6</td>
</tr>
</tbody>
</table>

Table 3: Worker accuracy on Task 1b, broken down by concept category. We see that some hard concepts cannot be easily disambiguated, e.g., Similar Animal. Some concepts we intentionally do not show to workers as an example in order to assess their ability to infer the given concept from a related example. These concepts are marked with an asterisk(*)

<table>
<thead>
<tr>
<th>Design</th>
<th>Similar Animal</th>
<th>Stuffed Toy*</th>
<th>Robot</th>
<th>Statue</th>
<th>Cartoon</th>
<th>Objects*</th>
<th>Unseen Ambiguities*</th>
<th>Small Dog Breed</th>
<th>Plane</th>
<th>Dog</th>
<th>Different Animal</th>
</tr>
</thead>
<tbody>
<tr>
<td>B0</td>
<td>37.0</td>
<td>22.2</td>
<td>33.3</td>
<td>18.5</td>
<td>62.2</td>
<td>74.1</td>
<td>88.9</td>
<td>88.9</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>B1</td>
<td>14.8</td>
<td>22.2</td>
<td>25.9</td>
<td>29.6</td>
<td>57.8</td>
<td>59.3</td>
<td>82.2</td>
<td>94.4</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>IMG</td>
<td>44.4</td>
<td>100.0</td>
<td>100.0</td>
<td>92.6</td>
<td>100.0</td>
<td>100.0</td>
<td>75.6</td>
<td>88.9</td>
<td>100.0</td>
<td>95.6</td>
<td>100.0</td>
</tr>
<tr>
<td>TAG</td>
<td>74.1</td>
<td>100.0</td>
<td>88.9</td>
<td>88.9</td>
<td>97.8</td>
<td>100.0</td>
<td>80.0</td>
<td>94.4</td>
<td>100.0</td>
<td>91.0</td>
<td>100.0</td>
</tr>
<tr>
<td>IMG+TAG</td>
<td>48.1</td>
<td>88.9</td>
<td>92.6</td>
<td>77.8</td>
<td>97.8</td>
<td>96.3</td>
<td>77.8</td>
<td>91.7</td>
<td>96.3</td>
<td>95.6</td>
<td>88.9</td>
</tr>
</tbody>
</table>

5.3.2 Learning Closely Related Concepts. To see if the crowd-workers learn closely related concepts without being explicitly shown examples, consider “Robot Dog” and “Stuffed Toy” as two types of a larger “Toy Dog” children’s toy concept. In Task 1b, the workers are shown the concept “Robot Dog” as an example labeled as NO, without being shown an example for “Stuffed Toy”.

Table 3 shows that workers learn the related concept “Stuffed Toy” and accurately label the instances that belong to this concept. The performance gain for the concept “Toy Dog” is the same as the gain for “Robot Dog”, when we compare design IMG+TAG and B1. Other similarly unseen concepts (marked with an asterisk(*) in the table) show that workers are able to learn the Requester’s intent for unseen concepts if given examples of other, similar concepts.

5.3.3 Limitations of aggregation and peer-agreement with Ambiguous Examples. It is not always possible or cost-effective to obtain expert/gold labels for tasks, so Requesters often rely on peer-agreement between workers to estimate worker reliability. Similarly, majority voting or weighted voting is often used to aggregate worker labels for consensus [20, 41, 47]. However, we also know that when workers have consistent biases, aggregation will amplify bias rather than mitigate it [21].

While we find agreement often correlates with accuracy, and so have largely omitted reporting it in this work, we do find a number of concepts for which the majority chooses a wrong answers, producing high agreement but low accuracy. Recall that our results are reported over 9 workers per example, whereas typical studies use a plurality of 3 or 5 workers. Also recall that Tasks 1b and 2a (Section 4.1) represent two of our less intuitive annotation tasks for which Requester intent may be at odds with worker intuition, requiring greater task clarity for over-coming worker bias.

Table 4 shows majority vote accuracy for these tasks for the baseline B1 design which (perhaps typical of many Requesters) includes illustrative examples but not necessarily the most informative ones. Despite collecting labels from 9 different workers, the majority is still wrong, with majority vote accuracy on ambiguous examples falling below 50%.

6 CONCLUSION AND FUTURE WORK
6.1 Summary and Contributions

While crowdsourcing has made it easy, fast, and cheap to collect labeled data, ensuring data quality remains a continuing challenge today. In adopting the new advantages crowdsourcing offers, researchers may have gone too far in “throwing out the baby with the bath water”, failing to incorporate best practices of traditional annotation into our new crowdsourcing workflows.

In this work, we have presented a three-stage FIND-RESOLVE-LABEL workflow as a novel mapping of traditional annotation processes, involving iterative refinement of guidelines by expert annotators, onto a light-weight, structured task design suitable for crowdsourcing. By careful task design and intelligently distributing effort between crowd vs. Requester, we have shown that the crowd can play a valuable role in reducing Requester effort while also
helping the Requester better understand their data and generate clearer task instructions for the crowd.

While including illustrative examples in instructions is known to be helpful [46], we have shown that not all examples are equally informative to annotators, and that intelligently selecting ambiguous corner-cases can improve labeling quality. Moreover, we see that the crowd performs worst on ambiguous instances and hence need the most help with such cases and less intuitive Requester intents, which run counter to internal biases of the annotators. In these cases, we can see high agreement between workers on answers contrary to what the Requester defines as correct. Such tasks are likely to produce the wrong label when we induce consensus using techniques like majority vote, and hence instruction clarity will be more critical in such cases.

Finally, we found that workers were able to infer the correct labels for closely related concepts, hence we do not need to exhaustively find and annotate all ambiguous concepts that could be encountered during the task. If we find sufficient examples, the crowd can infer the labels of unseen examples accurately.

### 6.2 Future Work

While we evaluate our strategy on an image labeling task, our approach is more general and could be usefully extended to other domains and tasks. For example, consider collecting document relevance judgments in information retrieval [3, 34, 39], where user information needs are often subjective, vague, and incomplete. Such generalization may raise new challenges. For example, while our task involves image search, we allow us to simply point workers to Google, in other domains, it might not be easy for the crowd to search the domain without the help of additional search tools.

Alonso [2] proposes having workers perform one task for the sole purpose of helping them learn the data task in order to perform better on another task of real interest. While we prohibited workers from performing multiple tasks for us in order to prevent such learning effects from confounding our analysis, it would also be interesting to take advantage of such learning. Workers performing Stage 1 ought to be expected to perform better in Stage 2.

Another best practice from LDC is deriving a decision tree for common ambiguous cases which annotators can follow as a principled and consistent way to determine the label of ambiguous examples [19]. How might we use the crowd to induce such a decision tree? Prior design work in effectively engaging the crowd in clustering [9] could likely also guide design here.

Both our Stage 1 filtered scenario and our Stage 2 RESOLVE task involved Requester selection of ambiguous examples. As such, there is an opportunity for reusing this Requester effort if both of these are employed. Examples selected in Stage 1 filtering could also filter the set of examples for Stage 2, or labeling could be performed during Stage 1 filtering to bypass Stage 2 altogether. Similarly, if the workflow is started over again to improve quality after Stage 3, Requester effort from the prior iteration could be reused.

More interesting still are directions for further reducing Requester effort. For example, the crowd could be called upon to verify and prune ambiguous examples collected in the initial FIND stage. Examples flagged as spam or assigned a low ambiguity rating could be automatically filtered out prior to Requester involvement in Stage 2. Crowd ambiguity ratings could also be used to rank examples to further guide Requester selection in Stage 2. A more ambitious and intriguing idea would be to see how well the crowd could guess Requester intent—under what conditions could we reduce or eliminate Requester involvement in determining the appropriate label for seemingly ambiguous cases?

### ACKNOWLEDGMENTS

We thank the reviewers for their valuable feedback in helping us further improve this work. We also thank our many talented crowd contributors, without whom our research would not be possible. This study is supported in part by National Science Foundation grant No. 1253413. Any opinions, findings, and conclusions or recommendations expressed by the authors are entirely their own and do not represent those of the sponsoring agencies.

### REFERENCES


