UT Austin in the TREC 2012 Crowdsourcing Track's Image Relevance Assessment Task

Hyun Joon Jung and Matthew Lease

School of Information
University of Texas at Austin
hyunJoon@utexas.edu ml@ischool.utexas.edu

Abstract. We describe our submission to the Image Relevance Assessment Task (IRAT) at the 2012 Text REtrieval Conference (TREC) Crowdsourcing Track. Four aspects distinguish our approach: 1) an interface for cohesive, efficient topic-based relevance judging and reporting judgment confidence; 2) a variant of Welinder and Perona's method for online crowdsourcing [17] (inferring quality of the judgments and judges during data collection in order to dynamically optimize data collection); 3) a completely unsupervised approach using no labeled data for either training or tuning; and 4) automatic generation of individualized error reports for each crowd worker, supporting transparent assessment and education of workers. Our system was built start-to-finish in two weeks, and we collected approximately 44,000 labels for about $40 US.

1 Introduction

Internet-based crowdsourcing is transforming traditional labor practices for many common data processing tasks [1,11]. To date, statistical quality assurances to crowdsourcing have predominantly focused on *offline* label aggregation approaches, where labels are first obtained and only aggregated afterward. While this permits a simple staged approach, with minimal coupling between label collection and aggregation stages it precludes dynamic optimization of labeling effort, meaning the same labeling effort is expended on all examples without regard to varying example difficulty or variable skill of the workforce. In contrast, Welinder and Perona proposed an *online* approach which integrates label collection with aggregation, dynamically evaluating both aggregated labels and workers as labels are collected [17]. While under some assumptions the online approach offers no benefit over offline [8], in general the greater flexibility of the online approach provides additional opportunities for optimization. We describe a variant of Welinder and Perona's approach which we apply to the IRAT task of collecting image relevance judgments. This enables us to reduce effort on simple examples, dynamically request additional labels for uncertain examples, and identify trusted workers.

As in Welinder and Perona's approach, we adopt a completely unsupervised method. While supervised methods tend to outperform unsupervised methods for a variety of tasks, including aggregation of noisy crowdsourcing labels, gold data required for supervised training is typically assumed to be expensive to create. This is particularly true in a crowdsourcing scenario, where supervised approaches require constant creation of

new gold labels to avoid common forms of online fraud based on worker memorization and sharing of gold answers (when gold is reused across runs instead of being renewed).

To collect labels via Amazon's Mechanical Turk service, we created a cohesive and cost-effective relevance judging interface which groups together a large set of images needing to be judged for the same search topic. As part of this interface, we also required that workers self-report judgment confidence; while we use this confidence data in our data collection process, primarily the confidence data still awaits further analysis.

To educate and retain trusted workers, we also experiment with a communication strategy which provides crowd workers with individualized error reports, as well as bonuses tied to a metric in these reports. The individualized error reports include a variety of performance measures and an illustrative visual plot. Again, substantive analysis of these reports and corresponding bonusing policy remains forthcoming.

The question of appropriate payment for crowdsourcing tasks remains complicated and unresolved. To date, the question has been framed predominantly in terms of how to optimize production (e.g., quality and speed relative to cost). While this simple framing is incomplete, even it presents a host of interacting concerns. A variety of crowdsourcing studies have explored such concerns [10,3], and a far larger body of work in other fields (e.g., psychology, economics, business) has considered the general question of appropriate payment beyond the crowdsourcing context. Beyond optimizing production, however, the questions raises other, broader considerations one must wrestle with, spanning ethics, economic sustainability, and legal regulation [7,14,18,1,4]. Like Irani et al., we do resolve these broader concerns but believe it important to acknowledge them to promote community awareness and provoke ongoing discussion.

2 Image Relevance Assessment Task (IRAT)

IRAT represents one of the two shared tasks in the 2012 TREC Crowdsourcing Track. The task requires collection of topical, binary relevance judgments for approximately 20K topic-image pairs spanning a total of 89 topics. 69 of these topics are taken from the ImageCLEF 2009 evaluation campaign, with 200 Belga images per topic needing to be judged. The remaining 20 topics come from the ImageCLEF 2012 evaluation campaign, with 300 MIRFLICKR images per topic to be judged. In sum, a total of 69*200 + 20*300 = 19,800 topic-image pairs required judging.

3 Relevance Judging Interface

Figure 1 shows our judging interface for an example topic and image set. We implemented this interface as an external Human Intelligence Task (HIT) hosted on our own webserver, allowing us greater design flexibility than possible using Amazon's predefined internal HIT widgets, with the additional benefit of avoiding fraudulent robot workers built for Amazon's standard interface widgets. Each HIT includes up to 100 images (25 rows of 4 columns) to be judged for a given topic. The topic is described in the top of the HIT page, and each image is displayed at the entry of the table which is composed of 4 columns by 25 rows.

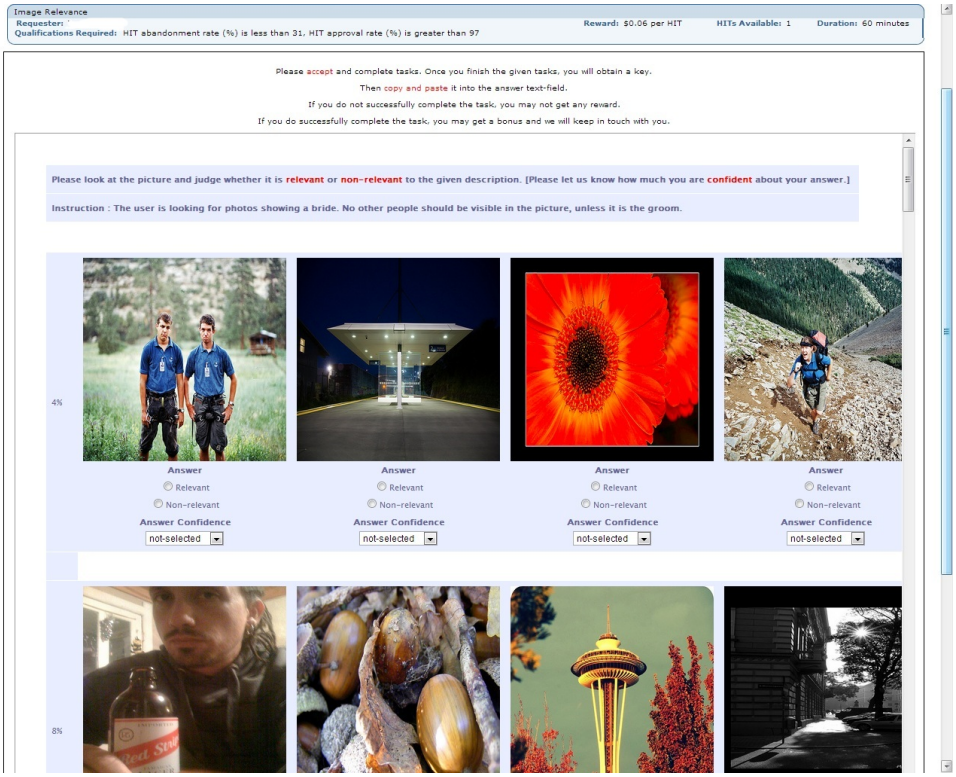


Fig. 1. Our judging interface for the image relevance assessment task. Each crowdsourcing task includes up to 100 images which need to be judged for a given search topic, whose description appears at top. Each judgment requires selecting the relevant or non-relevant option via a radio button, as well as reporting a likert-scale confidence score for each judgment via a drop-down list. Confidence options range from 5 (very confident) to 1 (very unsure). Both the radio button and drop-down list are initialized without a selection to avoid any possible bias in default selection.

Several years ago, we collected crowd judgments for the TREC 2010 Relevance Feedback Track [2,15,5]. For that task, we designed the relevance judging interface to display 5 documents at a time for a given topic, and we paid roughly a $0.01 per judgment. If we had reused this same design here for our 20K topic-image pairs, collecting only a single judgment per example would have cost roughly $200. For this task, however, we expected that judging image relevance would be easier, faster, and more fun for workers than making text-based relevance judgments, and many image thumbnails can easily be placed on the same page. As a result, our revised HIT design further optimized production for this task by using 100 images per page with $0.05 offered per HIT, or $0.01 per 20 judgments ($\frac{1}{20}$th the cost of the original design). While we cannot yet comment on judgment accuracy, we can report HITs were still accepted and completed quickly, with satisfied workers to the best of our knowledge.

Because many images could be easily accommodated on a single, cohesive task page, judges could also see a broad spectrum of images in making relative decisions

on relevance, and we expected (but have not yet evaluated) that this would reduce judging effort similar to prior studies showing pair-wise preference judging to be easier than making item-at-a-time absolute relevance judgments. Images could also be judged in any arbitrary order (e.g., one could simply judge images left-to-right, or first find and mark relevant images and then return to mark each remaining image as non-relevant). In addition to the common crowdsourcing question of how to determine optimal pay (according to some external performance criteria [10] along with other considerations [14,4]), it would be interesting to study how exposing judges to more or fewer images in parallel impacts satisfaction, effort, relative judgments, and the judges' own self-developed criteria for deciding topical relevance. We also wonder how workers' different judgment ordering or other assessment practices affect performance metrics.

In regard to design, we intentionally have no default setting for the radio selection of relevant vs. non-relevant in order to avoid any bias in default behavior and so that equal effort is required for selecting either option [9]. However, even such a simple design decision opens a variety of questions for further study. For example, with no default option, how does the extra effort involved affect task completion time, worker satisfaction (enjoyment, recruitment, and retention), task pricing, and/or quality of the actual relevance judgments. As a point of contrast, we note CrowdFlower's interface for content moderation (another binary judgment task) sets the dominant class as the default selection, thereby requiring workers to click only on examples of the less frequent class.

Raykar et al. [12] measured worker performance via sensitivity (true positive rate) and specificity (1-false positive rate). Following this, Raykar and Yu [13] proposed the following single "spammer" metric as a function of sensitivity and specificity: A novel aspect of our data collection in general is having workers self-report confidence of each judgment, which obviously has potential to further inform automatic analysis and aggregation of collected labels. This poses a variety of similar questions for further study. For example, collecting these confidence scores requires twice as much clicking for the worker, which introduces similar tradeoffs. How do we balance the tradeoff between simply collecting more judgments without confidence scores vs. collecting fewer judgments and requiring confidence self-reporting? Perhaps most critically, how informative and useful are the self-reported confidence scores once collected? While we make use of these confidence scores in our iterative data collection algorithm, this analysis remains for our future work.

4 Evaluating Workers and Establishing Trust

$$spammer(w) = \frac{\|recall(r) + specificity(s) - 1\|}{\sqrt{2}}$$

Figure 2 presents a visualization of this metric with additional explanation. We adopt this as our primary metric for evaluating worker performance, with a parameter minimum threshold for establishing trust. Because our approach is completely unsupervised, workers are evaluated with respect to "pseudo-gold" produced by aggregating judgments of other workers. A worker must also complete a minimum number of judgments on pseudo-gold to become trusted, a simple surrogate for a confidence interval

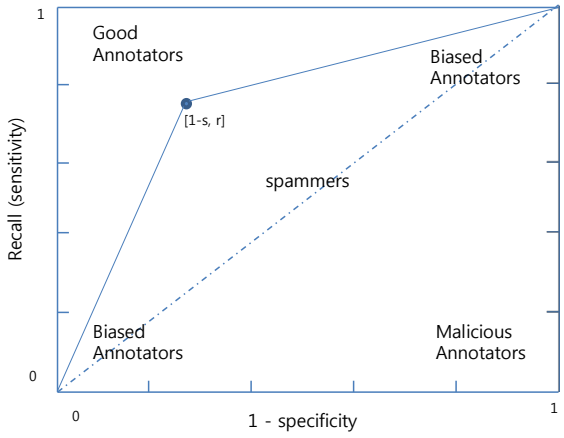


Fig. 2. Raykar and Yu's spammer metric for evaluating annotator performance (copied from [13]). As similarly noted by Ipeirotis [6], it is important to distinguish a worker's labeling errors arising from consistent (correctable) biases vs. errors representing unrecoverable noise. In general, we desire worker labels to be strongly correlated to true labels (either directly or inversely; in the latter case we merely flip worker predictions). In contrast, uncorrelated labels offer little value. Accuracy as a metric is also less meaningful when classes (e.g., binary relevant and non-relevant classes) are significantly imbalanced. For example, if only 10% of images are relevant, 90% accuracy can be achieved by simply labeling all images as non-relevant. In contrast with accuracy, Raykar's spammer metric is similar in spirit (though different in implementation) to measuring average recall across classes in order to limit the influence of the dominant class. A perfect worker would be located at [0,1], a perfectly inverse worker at [1,0], and constant workers (all examples assigned to the same class) at [0,0] and [1,1], respectively. The spammer score is defined as the distance from the diagonal $y = x$ to a worker's score $(1 - s, r)$, indicating degree of correlation between the worker's labels and the true class assignments.

to bound our measurement error in estimating the worker's spammer score based on his observed judgments.

The spammer score and trust also had two other consequences for a worker. Firstly, workers achieving a spammer score of at least 0.80 received a bonus explicitly and proportionately tied to that score. For example, a worker making 1,000 relevance judgments with spammer score of 0.80 earned a bonus of \$4 $(1,000 \times \$0.005 \times 0.80)$. Secondly, only trusted workers received personalized error reports, as shall be later discussed.

5 Online Crowdsourcing Algorithm

Figure 3 depicts our online data collection algorithm via a flow diagram. Our algorithm has two parts, as shown in the diagram: (a) label collection and (b) worker evaluation. We partition the set of examples E into several subsets of increasing size in order to

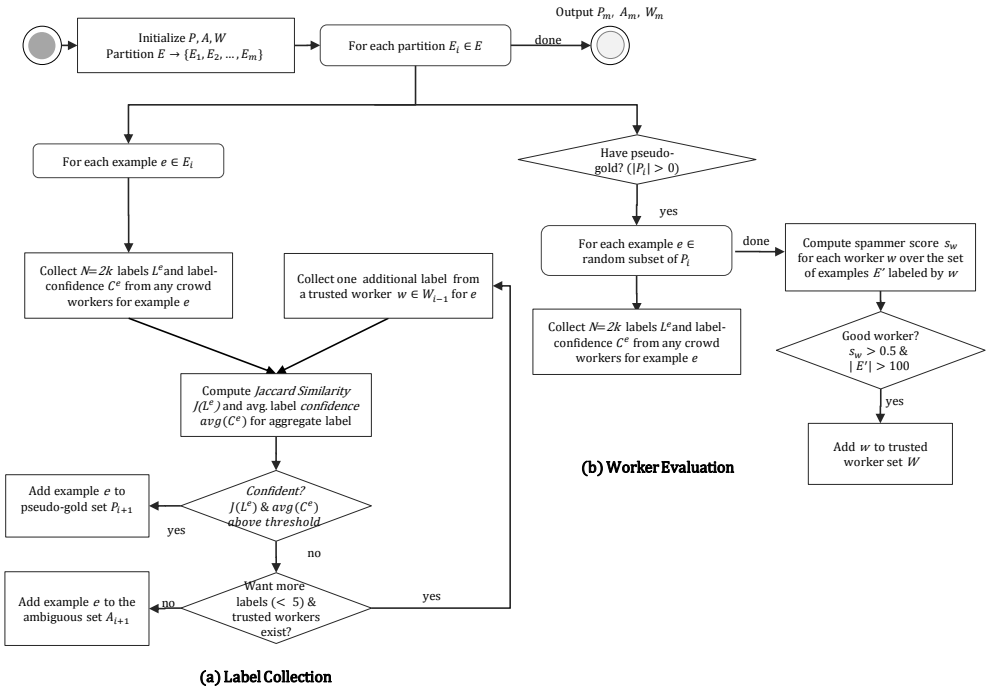


Fig. 3. A flow diagram depicts our unsupervised, iterative algorithm for online data collection. The set of input examples to be judged E (topic-image pairs) is partitioned into m ordered subsets of progressively larger size. Data is collected in stages corresponding to each partition, allowing later stages to benefit from trusted workers identified earlier. To begin each stage, $2k$ relevance judgments per example are collected via an open call to all crowd workers. If both the Jaccard agreement between collected labels and the average judgment confidence exceed parameter thresholds, the aggregate label is then accepted into the "pseudo-gold" set and no additional judgments are requested. However, if the label remains ambiguous and we have not exceeded our per-example labeling budget, we then iteratively: 1) collect an additional label via an open call to known, trusted workers; 2) perform the same test as above; and 3) repeat until the aggregate label is accepted or the labeling budget is exceeded. Trusted workers are identified using a random subset of pseudo-gold examples. For each pseudo-gold example selected, $2k$ additional labels are requested via an open call to any non-trusted workers. We then evaluate each worker's individual performance on the set of labeled examples given the pseudo-gold labels. If the worker judges a minimum number of examples and achieves a spammer score exceeding a parameter threshold, the worker is then promoted into the set of trusted workers. A labeling effort tradeoff exists between *exploration* to identify trusted workers (i.e., collecting more labels on pseudo-gold examples), and *exploitation* to collect more labels for ambiguous examples.

collect labels in a gradual and incremental way. The algorithm is initialized with three empty sets: pseudo-gold labels P, ambiguous examples A, and trusted workers W.

The process starts by obtaining an even number $(2k)$ of independent labels $L^e = \{l_1^e, l_2^e, ..., l_{2k}^e\}$ from crowd workers. Next, it measures the level of agreement $J(L^e)$ between obtained labels using the Jaccard similarity coefficient J computed between labels sets (the size of the intersection over the size of their union, as in [16]). In addition, we leverage self-reported judgment confidence scores reported by workers $C^e = \{c_1^e, c_2^e, ..., c_{2k}^e\}$.

If these two measures ($J(L^e)$, $avg(C^e)$) exceed predefined thresholds, the example e's labels are aggregated by simple majority voting and added to the pseudo-gold set P. Otherwise, the example e is considered ambiguous and more judgments are collected.

Specifically, we ask our pool of trusted workers for an additional label for the example. We then again test for agreement and promote the example to pseudo-gold if sufficient agreement is observed, and if not, continue to iterate until either agreement is reached or we exceed a annotation budget limit, in which case we give up on the example and output it in the ambiguous set.

The second part of the online algorithm shown in Figure 3(b) is worker evaluation. We measure the performance of a new worker w over the set of pseudo-ground truth labels which is constructed in the previous partition of examples. Thus, this process does not work in the first partition since no pseudo-ground truth is ready. When a pseudo-ground truth set P is not empty, it measures each new worker w over pseudo-ground truth labels in P. If a worker's performance exceeds a given threshold α_{per}, this worker is qualified for a trusted worker and added into the trusted worker pool W. This process iterates until a set of example partitions are empty or the number of obtained labels for each example exceeds a predefined threshold. Finally, it produces a set of pseudo-ground truth labels P_m along with a pool of trusted workers W_m.

6 Personalized Error Reports and Worker Education

Another unique aspect of our approach was the automatic generation of individualized error reports for each crowd worker. Figure 4 shows an example personalized error report which shows worker accuracy, precision, recall, specificity, and LAM (logistic average mis-classification), along with a ROC plot including Raykar and Yu's spammer score [13]. Error reports were implemented as an interactive Google Web App Component hosted on our webserver, which workers could inspect and explore to better understand how they were being evaluated and which of their judgments were automatically determined to be incorrect. Only trusted workers were sent these reports, as they were the workers we most wanted to train and retain, and with whom we hoped to foster relationships. We also wished to minimize risks of cheating by other workers, since gold answers disclosed in the error report were sometimes reused later.

These error reports raise a variety of research questions for which additional analysis is still needed. For example, the current reports are extremely dense with very technical information; how much of this is clearly understood by the workers, and how could we both better pare down what information is included and better convey this information to laymen? One worker wrote, "What's the meaning of this plot?" Another

worker wrote, "This one is useful to figure out what I've done." How much attention do workers give such reports and what parts of them are most useful? Finally, how do such error reports actually impact the bottom line (task performance, education and improvement, enjoyment, retention, etc.)?

Taking a step back, these error reports might be seen as a very simple form of educational assessment with workers as online students. Whereas crowd work today (particularly micro-work) predominantly involves simple tasks and/or using workers' existing skill sets rather than further developing workers' skills, we forsee a not-to-distant future in which online education and crowd work converge in exciting ways to deliver more scalable education and integrate vocational practice into educational curriculum [1]. After all, crowd work inherently involves training (learning) and assessment, as workers often need to acquire new skills to perform new tasks, before or in the midst of performing the actual work. Workers may also polish and refine existing skills while completing more familiar tasks. Requesters must continually engage in quality assurance. Such a training-assessment cycle of work offers potentially exciting synergies with online, education by-doing. For example, DuoLingo (duolingo.com) explores this direction for foreign language learning. This idea can be generalized much further; for example, content generation tasks could be designed to better assess and enhance writing skills. Tracking and mining of work history could support personalized instruction and feedback, as well as recommending new tasks and learning modules. As workers master new skills and are assessed, badges or credentials could document this proficiency so that others can recognize and utilize this enhanced skill set.

7 Evaluation

The set of all examples (topic-image pairs) was partitioned into four subsets of increasing size; 5%, 15%, 30% and 50%. In order for an aggregate label to be accepted by the system as pseudo-gold, we used a Jaccard similarity coefficient of $J >= 0.67$ and average judgment-confidence score of 4.0. With regard to worker evaluation, if a worker's spammer score $s(w) >= 0.5$ over at least 100 judgments, the worker was promoted to trusted status (the threshold 0.5 indicates that a sum of recall r and specificity s should be approximately 1.7.

Figure 5 (a) shows the variations of number of pseudo-ground truth labels vs. ambiguous examples over four partitions of the example set in the experiments. The ratios of ambiguous examples across the partitions are very similar regardless of the size of examples in the each step. It varies from 18.7% to 21.5%, which indicates that each step does not affect the ratio of ambiguous examples over steps in this experiment. Next Figure 5 (b) presents the increase of the number of trusted workers over the partitions of the example set. The number of trusted workers increased over the steps based on the threshold (α_{spam} = 0.5). The ratios of trusted workers vs. all workers across the steps vary from 21% ~28%.

The proposed algorithm obtains the different number of labels for each example until the predefined threshold is achieved. Figure 6 shows the distribution of labels collected per example, showing 80% of examples were accepted as pseudo-gold after

only two judgments. Only 1% of examples required more than three judgments before being accepted as pseudo-gold.

We did not explicitly request worker feedback given our short timeframe but did receive some anecdotal comments which are mildly suggestive of worker satisfaction: "I like this one", "Good job", "Great". In addition to the worker comments on the error report mentioned earlier, another worker asked, "How do I get a bonus?", suggesting the bonusing policy did attract interest but still wasn't sufficiently clear.

Finally, we give a breakdown of the total cost of our data collection. As discussed earlier, each of our HITs includes 100 images, and we pay \$0.05 per HIT (excluding Amazon's 10% overhead rate, which raises actual cost to \$0.055 per HIT). \$22 was paid for the label collection ($44,000\ labels/100 \times 0.05$) , and \$5 was paid for worker evaluation to identify trusted workers ($10,000\ labels/100 \times 0.05$). Finally, we awarded a bonus to 4 trusted workers totaling \$10 based on our bonusing policy (judgments per worker w $\times$ \$0.005 $\times$ spammer score $s(w)$). In total, \$22 + \$5 + \$10 = \$37.

8 Conclusion

Our approach was characterized by four factors: 1) an interface for cohesive, efficient topic-based relevance judging and reporting judgment confidence; 2) a variant of Welinder and Perona's method for online crowdsourcing [17]; 3) a completely unsupervised approach; and 4) automatic generation of individualized error reports for each crowd worker. Our prototype system was built start-to-finish in two weeks, approximately 44,000 labels were collected for about \$40 US, and the experience helped us to identify many interesting research questions which will be fruitful for further study.

9 Acknowledgments

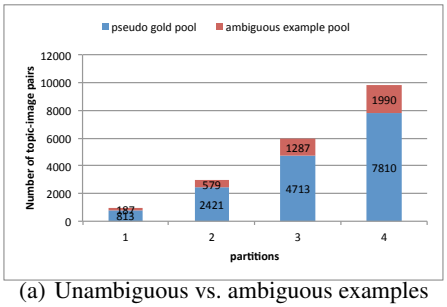
We thank Amazon.com for their continuing sponsorship of the TREC Crowdsourcing Track, supporting our use of Mechanical Turk. The authors' recent industrial experiences at Intelius and CrowdFlower, respectively, also informed our approach. This work was partially supported by DARPA Young Faculty Award No. N66001-12-1-4256, a Yahoo! Faculty Research and Engagement award, and a Temple Fellowship. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of their funding agencies.

References

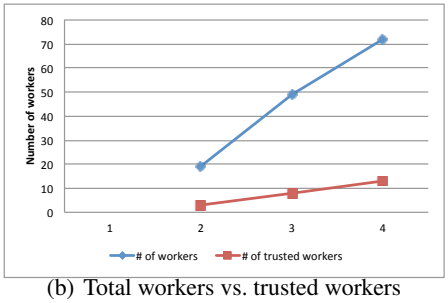
1. J. N. Aniket Kittur, M. S. Bernstein, E. Gerber, A. Shaw, J. Zimmerman, M. Lease, and J. J. Horton. The future of crowd work. In *ACM Conference on Computer Supported Cooperative Work (CSCW)*, February 2013. Social Science Research Network (SSRN) ID: 2190946.
2. C. Buckley, M. Lease, and M. D. Smucker. Overview of the TREC 2010 Relevance Feedback Track (Notebook). In *The Nineteenth Text Retrieval Conference (TREC 2010) Notebook*, 2010.
3. S. Faridani, B. Hartmann, and P. Ipeirotis. Whats the right price? pricing tasks for finishing on time. In *Proc. of AAAI Workshop on Human Computation*, 2011.

4. K. Fort, G. Adda, and K. Cohen. Amazon Mechanical Turk: Gold mine or coal mine? *Computational Linguistics*, 37(2):413–420, 2011.
5. C. Grady and M. Lease. Crowdsourcing Document Relevance Assessment with Mechanical Turk. In *Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 172–179, 2010.
6. P. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67, 2010.
7. L. Irani and M. Silberman. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceeding of the Annual ACM SIGCHI Conference on Human Factors in Computing Systems*, 2013.
8. D. Karger, S. Oh, and D. Shah. Budget-optimal task allocation for reliable crowdsourcing systems. *arXiv preprint arXiv:1110.3564*, 2011.
9. A. Kittur, E. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proceeding of the 26th Annual ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 453–456, 2008.
10. W. Mason and D. Watts. Financial incentives and the performance of crowds. *ACM SIGKDD Explorations Newsletter*, 11(2):100–108, 2010.
11. A. Quinn and B. Bederson. Human computation: a survey and taxonomy of a growing field. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 1403–1412. ACM, 2011.
12. V. Raykar, S. Yu, L. Zhao, G. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *Journal of Machine Learning Research*, 99:1297–1322, 2010.
13. V. C. Raykar and S. Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *Journal of Machine Learning Research*, 13:491–518, 2012.
14. M. Silberman, L. Irani, and J. Ross. Ethics and tactics of professional crowdwork. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):39–43, 2010.
15. W. Tang and M. Lease. Semi-supervised consensus labeling for crowdsourcing. In *Proceedings of the ACM SIGIR 2nd Workshop on Crowdsourcing for Information Retrieval (CIR)*, 2011.
16. E. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management*, 36(5):697–716, 2000.
17. P. Welinder and P. Perona. Online crowdsourcing: Rating annotators and obtaining cost-effective labels. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 25–32, 2010.
18. S. Wolfson and M. Lease. Look before you leap: Legal pitfalls of crowdsourcing. In *Proceedings of the 74th Annual Meeting of the American Society for Information Science and Technology (ASIS&T)*, 2011.

Fig. 4. A screenshot of an example individualized error report of a given worker's performance (the worker's ID is intentionally occluded in the screenshot). Such error reports are automatically generated for each worker and output as an interactive Google Web App Component that is hosted on our webserver. The top-left panel shows a set of numerical metrics characterizing the worker's performance. The top-right panel shows a ROC plot (recall vs. 1-specificity) visualization. The bottom panel provides a listing of the worker's specific mistakes (based on pseudo-gold aggregate labels. This listing promotes transparent performance assessment and allows workers to report any errors they find. Requesters can easily browse performance across individual workers as well as aggregate statistics. Workers are notified of the URL to their individual error reports via the Mechanical Turk API, though to avoid subsequent cheating on disclosed gold, only trusted workers are notified. Additional security measures are needed to ensure each worker's error report is only accessible to the given worker.



(a) Unambiguous vs. ambiguous examples



(b) Total workers vs. trusted workers

Fig. 5. The stacked bar plot in Figure (a) (left) shows the relative ratio of examples requiring only the minimal two judgments to achieve "pseudo-gold" status vs. "ambiguous" examples requiring more than two judgments to resolve ambiguity in each stage of data collection. Similarly, line plots in Figure (b) (right) show the relative ratio of trusted workers vs. all workers per stage. Since pseudo-gold from Stage 1 is used to identify trusted workers for Stage 2, no trusted workers exist in Stage 1 (total workers for Stage 1 are only omitted due to lack of trusted workers for contrast).

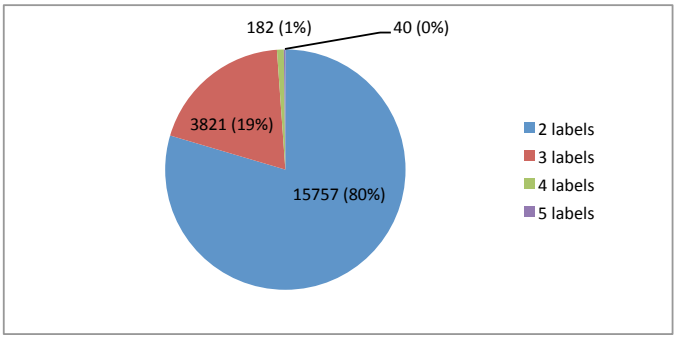


Fig. 6. The distribution of judgments collected per example. For 80% of examples, two judgments suffice to establish the aggregate label. Other examples were labeled iteratively until either accepted as pseudo-gold (i.e., achieving high agreement between judges and judgment confidence scores) or until the per-example maximum label budget of 5 was reached. Only 1% of examples required more than three judgments before being accepted as pseudo-gold.