Quality Management on Amazon Mechanical Turk

Panagiotis G. Ipeirotis
panos@stern.nyu.edu

Foster Provost
fprovost@stern.nyu.edu

Jing Wang
jwang5@stern.nyu.edu

Department of Information, Operations, and Management Sciences
Leonard N. Stern School of Business, New York University

ABSTRACT

Crowdsourcing services, such as Amazon Mechanical Turk, allow for easy distribution of small tasks to a large number of workers. Unfortunately, since manually verifying the quality of the submitted results is hard, malicious workers often take advantage of the verification difficulty and submit answers of low quality. Currently, most requesters rely on redundancy to identify the correct answers. However, redundancy is not a panacea. Massive redundancy is expensive, increasing significantly the cost of crowdsourced solutions. Therefore, we need techniques that will accurately estimate the quality of the workers, allowing for the rejection and blocking of the low-performing workers and spammers.

However, existing techniques cannot separate the true (unrecoverable) error rate from the (recoverable) biases that some workers exhibit. This lack of separation leads to incorrect assessments of a worker's quality. We present algorithms that improve the existing state-of-the-art techniques, enabling the separation of bias and error. Our algorithm generates a scalar score representing the inherent quality of each worker. We illustrate how to incorporate cost-sensitive classification errors in the overall framework and how to seamlessly integrate unsupervised and supervised techniques for inferring the quality of the workers. We present experimental results demonstrating the performance of the proposed algorithm under a variety of settings.

1. INTRODUCTION

Crowdsourcing has emerged over the last few years as an important new labor pool for a variety of tasks[3], ranging from micro-tasks on Mechanical Turk to big innovation contents conducted by Netflix and Innocentive. Amazon Mechanical Turk today dominates the market for crowdsourcing small tasks that would be too repetitive and too tedious for an individual to accomplish. Amazon Mechanical Turk established a marketplace where requesters can post tasks and workers complete them for relatively small amounts of money. Image tagging, relevance feedback, document labeling, are all tasks that are now routinely being completed online using the Amazon Mechanical Turk marketplace, delivering higher speed of completion and lower price than in-house solutions.

KDD-HCOMP'10, July 25, 2010, Washington DC, USA.

EXAMPLE 1. *Consider the following document labeling task: A worker has to look at a web site and decide whether there is any adult content on the page. The worker has to classify the page into one of the four categories:* G *for no adult content,* PG *for content that requires parental guidance,* R *for content that is appropriate mainly for adults, and* X *for hardcore porn. The rate of completion, when done by a trained intern, was 250 web sites per hour, at a cost of $15/hr. When posted on Amazon Mechanical Turk, the labeling rate went up to 2,500 web sites per hour and the overall cost remained the same. In other words, the use of a highly distributed workforce that works in parallel, allows a tenfold improvement in productivity.* □

Unfortunately, distributing labeling work to crowdsourcing platforms, such as Amazon Mechanical Turk, exposes the requester to quality risks. Verifying the quality of every submitted answer is an expensive operation and negates many of the advantages of crowdsourcing: the cost and time for verifying the correctness of the submitted solutions is typically comparable to the cost and time for performing the task itself. A common solution to this challenge is to rely on redundancy and repeated labeling: the same task is completed by multiple workers. Then using rules, such as majority voting, it is feasible to identify the correct answers. Unfortunately, repeated labeling is rather costly: if we ask 10 workers to complete the same task, then the cost of crowdsourcing solutions tends to be comparable to the cost of in-house solutions.

A solution is to use redundancy not only for identifying the correct answers to each task but also for measuring the labeling quality of the workers. Dawid and Skene [2] proposed a solution, based on an expectation maximization algorithm. The algorithm iterates until convergence, following two steps: (1) estimates the correct answer for each task, using labels assigned by multiple workers, accounting for the quality of each worker; and (2) estimates the quality of the workers by comparing the submitted answers to the inferred correct answers. The final output of the Dawid&Skene algorithm is the set of (estimated) correct answers for each task and a "confusion matrix" for each worker, listing the error probabilities for each worker. From the confusion matrix we can directly measure the overall error rate for each worker as the sum of the non-diagonal elements of the confusion matrix (properly weighted by the priors): this results in a single, scalar value as the quality score for each worker.

Unfortunately, the error rate alone is not sufficient to measure the inherent value of a worker. For example, workers may be careful but biased. In the web site labeling example, parents with young children tend to be more conservative than

others, and tend to classify PG-rated sites as R-rated sites, and R-rated sites as X-rated. Such workers give *consistently and predictably incorrect* answers. Unlike with spammers, with biased workers it is possible to "reverse" the errors and get a label assignment of much higher quality. So, in the presence of systematic bias, the naive measurement of error rate results in underestimates of the true quality of the worker and in potential incorrect rejections and blocks of legitimate workers.

As the main contribution of this paper, we present an algorithm that separates the unrecoverable error rate from bias. The basic idea is to use the confusion matrix of each worker and transform every assigned "hard" label from the worker into a "soft" label, which reflects the error rate of the worker. Then we can measure the uncertainty and cost associated with each such "soft" label. The final result is a quality score for each worker, potentially adjusted for the different costs of the misclassification errors, which measures the non-recoverable error rate of each worker. Our experiments illustrate that our algorithm is efficient in estimating the true quality of the workers.

2. ESTIMATING WORKER QUALITY

Dawid and Skene [2] presented an expectation maximization algorithm, using maximum likelihood, for inferring the error rates of annotators that assign class labels to objects, when the "gold" truth is unknown. Bayesian versions of the algorithm were recently proposed by Raykar et al. [4] and by Carpenter [1].

The EM algorithm of Dawid and Skene takes as input a set of N objects, $o_1, \ldots, o_N$, each being associated with a *latent* true class label $T(o_n)$, picked from one of the L different labels. Each object is annotated by one or more of the K workers, each having a varying degree of quality. To measure the quality of each worker, the algorithm endows each worker (k) with a *latent* "confusion matrix" $\pi_{ij}^{(k)}$, which gives the probability that worker (k), when presented with an object of true class i, will classify the object into category j.

Algorithm 1 presents a sketch of the process. The algorithm iterates between estimating the correct labels $T(o_n)$ for each of the objects, and estimating the error rates $\pi_{ij}^{(k)}$ for each worker. (The complete details are given in [2] and are rather secondary at this point.)

What is of interest to for our work, are the estimated error rates $\pi_{ij}^{(k)}$ for each worker and how we can use these values for estimating the non-recoverable error rate of each worker. So far, when estimating the quality of the workers, the main idea was to sum the non-diagonal entries of the matrix $\pi^{(k)}$ (i.e., sum all the values $\pi_{ij}^{(k)}$ with $i \neq j$), weighting each error rate using the estimated prior of each class. Unfortunately, this is insufficient. For example, consider the following example:

EXAMPLE 2. *Consider two workers that label web sites into two classes:* porn *and* notporn. *Worker A is* always *incorrect: labels all* porn *web sites as* notporn *and vice versa. Worker B classifies all web sites, irrespectively of their true class, as* porn. *Which of the two workers is better? A simple error analysis indicates that the error rate of worker A is 100%, while the error rate of worker B is "only" 50%.*[1] *However, it is easy to see that the errors of worker A are easily reversible, while the errors of worker B are irreversible. In fact, worker A is a perfect worker, while worker B is a spammer.* □

[1]Assume, for simplicity, equal priors for the two classes.

Input: Labels $l[k][n]$ from worker (k) to object o_n,
Output: Confusion matrix $\pi_{ij}^{(k)}$ for each worker (k), Correct labels $T(o_n)$ for each object o_n, Class priors $Pr\{C\}$ for each class C

1 Initialize error rates $\pi_{ij}^{(k)}$ for each worker (k) (e.g., assume each worker is perfect);
2 Initialize correct label for each object $T(o_n)$ (e.g., using majority vote);
3 **while** *not converged* **do**
4 Estimate the correct label $T(o_n)$ for each object, using the labels $l[\cdot][n]$ assigned to o_n by workers, weighting the votes using the error rates $\pi_{ij}^{(k)}$;
5 Estimate the error rates $\pi_{ij}^{(k)}$, for each worker (k), using the correct labels $T(o_n)$ and the assigned labels $l[k][n]$;
6 Estimate the class priors $Pr\{C\}$, for each class C;
7 **end**
8 **return** *Estimated error rates* $\pi_{ij}^{(k)}$, *Estimated correct labels* $T(o_n)$, *Estimated class priors* $Pr\{C\}$

Algorithm 1: The EM algorithm for worker quality estimation.

So, naturally a question arises: How can we separate low-quality workers from high-quality, but biased, workers? We examine this question next.

3. SEPARATING ERROR AND BIAS

After running the EM algorithm, we have some reasonably accurate estimates of the error rates $\pi_{ij}^{(k)}$ for each worker. How can we estimate from these values the intrinsic, non-recoverable error rate?

We start with the following observation: Each worker assigns a "hard" label to each object. Using the error rates for this worker, we can transform this assigned label into a "soft" label, which is the best possible estimate that we have for the *true* label assignment. So, if we have L possible classes and the worker assigns class j as a label to an object, we can transform this "hard" assigned label into the "soft" label:

$$\left\langle \pi_{1j}^{(k)} \cdot Pr\{C=1\}, \ldots, \pi_{Lj}^{(k)} \cdot Pr\{C=L\} \right\rangle \quad (1)$$

where $\pi_{ij}^{(k)}$ is the probability that worker (k) classifies into class j, an object that in reality belongs to class i, $Pr\{C=i\}$ is the prior that the object will belong to class i. We should note that the quantities above need to be normalized by dividing them with

$$Pr\{AC=j\} = \sum_{i}^{L} \pi_{ij}^{(k)} \cdot Pr\{C=i\} \quad (2)$$

the probability that worker (k) assigns label j to any object.

EXAMPLE 3. *Take the case of worker A from the previous example. When this worker assigns a label of* Porn *(assume that porn is class 1), then the corresponding soft label has all the "probability mass" in the* NotPorn *category:*

$$\underbrace{\begin{pmatrix} 1 \\ 0 \end{pmatrix}}_{\text{Assigned: Porn}} \Rightarrow \underbrace{\begin{pmatrix} 0 \\ 1 \end{pmatrix}}_{\text{Corrected to: NotPorn}}$$

On the contrary, for worker B, who always assigns porn, *the corresponding corrected soft label does not give us any information; the soft label simply says that the best guess are simply the class priors:*

$$\underbrace{\begin{pmatrix} 1 \\ 0 \end{pmatrix}}_{\textit{Assigned: Porn}} \Rightarrow \underbrace{\begin{pmatrix} Pr\{C=1\} \\ Pr\{C=2\} \end{pmatrix}}_{\textit{Corrected to: Class priors}}$$

□

So, what can we do with these soft labels? The basic idea is to estimate the *expected cost of a soft label*. To estimate the cost of a soft label, we need to consider the misclassification costs. In the simplest case, we have a cost of 1 when an object is misclassified, and 0 otherwise. In a more general case, we have a cost c_{ij} when an object of class i is classified into category j.

LEMMA 1. *Given the classification costs c_{ij} and a soft label $\mathbf{p} = \langle p_1, p_2, \ldots, p_L \rangle$, the expected cost of the soft label $\mathbf{p}$ is:*

$$Cost(\mathbf{p}) = \sum_{i=1}^{L} \sum_{j=1}^{L} p_i \cdot p_j \cdot c_{ij} \tag{3}$$

□

The proof is rather simple. The expected classification cost is equal to the probability of classifying the object in class i (which is p_i), multiplied by the probability of the object belonging to class j in reality (which is p_j), multiplied with the associated cost of classifying an object of class j into class i (which is c_{ji}). Summing across all classes, we have the result above.

The results illustrate that workers with error rate matrices that generate "soft" labels with probability mass concentrated into a single class (i.e., certain "posterior" labels) will tend to have low estimated cost, as the product of Equation 3 will be close to 0. On the contrary, workers that tend to generate "soft" labels that are spread out across classes (i.e., uncertain "posterior" labels) will tend to have high associated costs.

EXAMPLE 4. *Consider the costs for the workers A and B from the previous examples. Assuming equal priors across classes, and $c_{ij} = 1$, if $i \neq j$ and $c_{ij} = 0$, if $i = j$, we have the following: The cost of worker A is 0, as the soft labels generated by A are $\langle 0, 1 \rangle$ and $\langle 1, 0 \rangle$. For worker B, the cost is 0.5 (the maximum possible) as the soft labels generated by B are all $\langle 0.5, 0.5 \rangle$ (i.e., highly uncertain).* □

Given that we know how to compute the estimated cost for each label, we can now easily estimate the expected cost for each worker. We first compute the priors $Pr\{AC = i\}$ (see Equation 2), which is the prior probability of the worker assigning label i to an object. Then we compute the "soft label" that corresponds to the assigned label (see Equation 1). Given the soft label, we use Equation 3 to compute its expected cost. Now, knowing how often the worker assigns a label and the expected cost, we can compute the average misclassification cost of each worker. Algorithm 2 illustrates the process.

As expected, perfect workers will have a cost of zero and random workers or spammers will have high expected costs. Notice, as illustrated in the example above, that it is not necessary for a worker to return the correct answers in order to have low costs! As long as the errors are predictable and reversible, the worker is assigned a low expected cost.

This tends to resolve quite a few issues with online workers that exhibit systematic biases in their answers but also put a lot of effort in coming up with the answers. Prior approaches that relied on agreement generate a significant number of rejections for such workers, which in turn alienates such high-quality workers, and discourages them from working with employers that rely on worker agreement. The proposed algorithm alleviates these concerns.

Input: Error rates $\pi_{ij}^{(k)}$ for each worker, Misclassification costs $c[i][j]$, Class priors $Pr\{C\}$
Output: Expected for each worker

```
foreach Worker (k) do
    Estimate how often the worker (k) assigns label l
    (Pr{AC = l}), using Eq. 2;
    Cost[k] = 0;
    foreach Label l, assigned with probability Pr{AC = l} do
        Using Eq. 1, compute the soft label soft^(k)(l) that
        corresponds to label l assigned by worker (k);
        Using Eq. 3, compute Cost(soft^(k)(l)) for the soft
        label;
        Cost[k] += Cost(soft^(k)(l)) · Pr{AC = l};
    end
end
return Cost[k] for each worker (k)
```

Algorithm 2: Estimating the Expected Cost of each Worker

One question that may arise is how to estimate the baseline cost for a "random" worker. An easy baseline is to assume that a worker assigns as label the class with the maximum prior probability. Alternatively, if we assume a more strategic spammer, we can set as baseline the expected cost of a worker that *always* assigns the *same* label, and this label assignment results in the minimum expected cost. Both approaches result in a good threshold for cutting off bad workers.

4. EXPERIMENTS

We conducted an extensive set of experiments, in order to examine how accurately the proposed algorithm can estimate:

- the true quality of the labelers, and
- the true class of the examples.

Synthetic Experiments: We conducted a set of synthetic experiments, in order to have the flexibility of controlling the composition of the data set and the quality of the workers. First, we created a data set with 1000 objects, classified with equal proportions to four categories. Then, we generated 200 labelers with various degrees of classification accuracy, ranging from 0.35 (slightly above the baseline of 0.25) to 0.8, for each of the four categories. We also applied a bias/disturbance for 30% of the workers, by randomly selecting two classes and swapping the assigned labels.

Figure 4 shows how accurately the algorithm can estimate the "true quality" of a labeler, under a variety of settings. We noticed that, for quality estimation purposes, the number of labels assigned by a worker, is an important factor for the estimating accurately the true quality of the worker. In general, we need 20-30 labels from each worker (and approximately 5 labels per example) in order to estimate with high degree of accuracy the true quality of the worker. We also noticed that the number of labels per worker affects significantly the estimation quality, while the number of labels per object has a much smaller effect.

Figure 4 shows the average classification error for our algorithm, when detecting the true class for each example, for varying the number of labels per worker, and the number of labels per object. We observed that the number of labels per worker tends to affect only mildly the accuracy of the

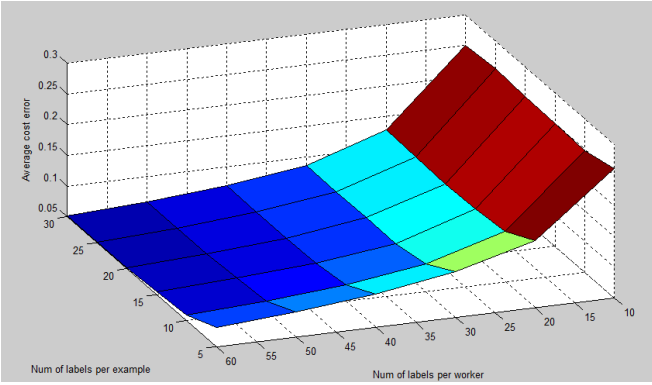

Figure 1: The estimation error when estimating the cost of each worker, under 0/1 classification error costs, with 4 categories of equal priors. 200 labelers with classification accuracy ranging uniformly from 0.35 to 0.8, and with a bias factor for 30% of the workers.

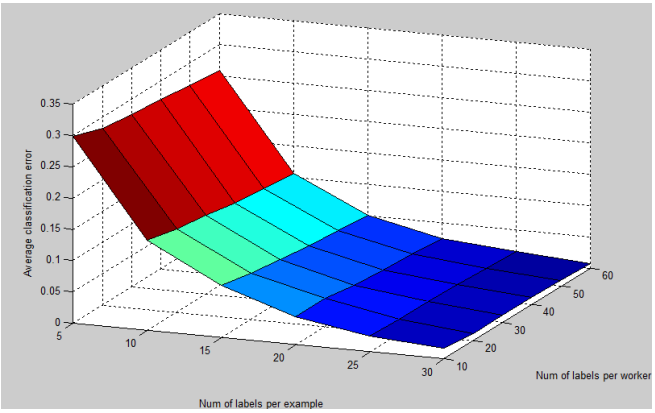

Figure 2: The average classification error under 0/1 classification error costs, with 4 categories of equal priors. 200 labelers with classification accuracy ranging uniformly from 0.35 to 0.8, and with a bias factor for 30% of the workers.

estimation. On the other hand, the effect of the number of labels per example was much more significant.

Experiments with Real Data: We also examined the effect of our algorithm for identifying spammers in a real-life task executed on Mechanical Turk. On this task, Mechanical Turk workers had to look at web pages and classify them into four categories (G, PG, R, X), depending on the presence of adult content on the page. (See Example 1.) We labeled 500 web pages, getting 100 labels for each web page. In total, 339 workers participated in the experiment.

To examine the effect of our algorithm, we executed our algorithm and measured the expected cost of each worker, using both our algorithm and the "naive" cost estimation using directly the confusion matrix of each worker. Then, we eliminated the workers that had an estimated cost of 0.5 and above. (A completely random worker generates a cost of 0.75, given that we have 4 categories.)

Our measurements indicated that our algorithm decreased the cost of annotation by 30%, while at the same time *increasing* the quality of annotation from 0.95 to 0.998. In other words, spammers generated 30% of the submitted answers, and decreased the quality of the results. The same process, using the "vanilla" EM algorithm, decreased the cost by only 1% and the quality remained effectively unchanged. The main reason for this failure is the inability of the EM algorithm to identify the "strategic" spammers; these sophisticated spammers identify the class with the highest class prior (in our case, the "G" class) and label all the pages as such. Our cost algorithm takes into consideration the fact that these "G" labels are very uncertain, while the naive approach does not penalize the spammer for the "correct" G labels, which are unfortunately uninformative.

We expect the results to be much more dramatic in terms of quality when we have significantly fewer labels per example. In such cases, a few strategic spammers can completely destroy the quality of the data, and the presence of accurate cost estimation techniques is paramount.

5. CONCLUSIONS

We presented an algorithm for quality management of the labeling process on crowdsourced environments. The algorithm can be applied when the workers should answer a multiple choice question to complete a task. The main novelty of the proposed approach is the ability to assign a single scalar score to each worker, which corresponds to the quality of the assigned labels. The score separates the intrinsic error rate from the bias of the worker, allowing for more reliable quality estimation. This also leads to more fair treatment of the workers.

Our experimental results indicate that the algorithm becomes accurate with approximately 5 labels per example, and by asking each worker to label at least 20 to 30 objects (preferably 50). Given that multiple choice questions take typically only a few seconds this is a rather easy requirement for a large number of tasks.

We should also note that the algorithm can seamlessly integrate the existence of "gold" data for learning the quality of labelers. It is a trivial change in the algorithm (i.e., do not update the true class of the "gold" examples, in Step 4 of Algorithm 1) and tends to improve significantly the estimation accuracy in the face of sparse data.

The code is open source and available at http://code.google.com/p/get-another-label/ and a demo is publicly accessible at http://qmturk.appspot.com/.

6. ACKNOWLEDGMENTS

Panagiotis G. Ipeirotis and Jing Wang were supported by the National Science Foundation under Grant No. IIS-0643846.

Bibliography

[1] Carpenter, B. Multilevel bayesian models of categorical data annotation. Available at http://lingpipe-blog.com/lingpipe-white-papers/, 2008.

[2] Dawid, A. P., and Skene, A. M. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics 28*, 1 (Sept. 1979), 20–28.

[3] Malone, T. W., Laubacher, R., and Dellarocas, C. Harnessing crowds: Mapping the genome of collective intelligence. Available at http://ssrn.com/abstract=1381502, 2010.

[4] Raykar, V. C., Yu, S., Zhao, L. H., Valadez, G. H., Florin, C., Bogoni, L., and Moy, L. Learning from crowds. *Journal of Machine Learning Research 11* (Apr. 2010), 1297–1322.