RELIABLE CLASSIFICATION BY UNRELIABLE CROWDS

Aditya Vempaty,⋆ Lav R. Varshney,† and Pramod K. Varshney⋆

⋆ Department of Electrical Engineering and Computer Science, Syracuse University
† IBM Thomas J. Watson Research Center

ABSTRACT

We consider the use of error-control codes and decoding algorithms to perform reliable classification using unreliable and anonymous human crowd workers by adapting coding-theoretic techniques for the specific crowdsourcing application. We develop an ordering principle for the quality of crowds and describe how system performance changes with the quality of the crowd. We demonstrate the effectiveness of the proposed coding scheme using both simulated data and real datasets from Amazon Mechanical Turk, a crowdsourcing microtask platform. Results suggest that good codes may improve the performance of the crowdsourcing task over typical majority-vote approaches.

***Index Terms*—** crowdsourcing, classification, error-control codes

1. INTRODUCTION

With the widespread growth of inexpensive computing and networking infrastructures, crowdsourcing has emerged as a new paradigm for human participation in distributed inference tasks [1–3]. Although both crowdsourcing and conventional team decision making [4] involve human decision makers, there are three major differences. First, the number of participants involved in crowdsourcing may be large. Second, contrary to typical mathematical models of traditional team decision making [8], members of the crowd may be unreliable or malicious [5–7] especially since they are often anonymous [9, 10]. Third, workers may not have sufficient domain expertise to perform full classification and may only be able to make simpler binary distinctions [11].

Hence achieving reliable crowdsourcing of difficult inference tasks raises novel signal processing questions. Ensuring reliable signal processing in the face of unreliable or malicious workers has previously been considered for numerical analysis [12] and binary classification tasks [13, 14]. In [13, 14], the authors consider the problem of task allocation in a crowdsourcing system. An iterative algorithm based on belief propagation was proposed for inferring the final answer from the workers' responses. This algorithm is shown to perform same as the best possible algorithm. They also provide numerical results for a large system size and show that their approach outperforms the majority approach. In contrast, we consider crowdsourcing M-ary classification tasks, such as multi-class object recognition from images into fine-grained categories [11]. Our aim in this work is to design the system to minimize misclassification probability by using a minimum Hamming distance decoder. In particular, we consider the use of codes for distributed classification that were originally developed in the context of large sensor networks to maximize classification performance while maintaining fault tolerance [15, 16]. We demonstrate the efficacy of this coding approach using simulations and through real data from Amazon Mechanical Turk [17], a paid crowdsourcing microtask platform. An ordering principle for the quality of crowds is also developed.

Sec. 2 develops a mathematical model of the crowdsourcing problem and Sec. 3 derives error performance expressions for both coding- and majority-based approaches. Sec. 4 introduces an ordering principle for quality of crowds based on mean worker quality and demonstrates through examples that systems with good codes outperform systems that use majority vote. Experimental results using real data from Amazon Mechanical Turk are provided in Sec. 5 and Sec. 6 concludes the paper.

2. CODING FOR CROWDSOURCING

Consider an image with an object to be classified into one of M fine-grained categories. Since object classification is often difficult for machine vision algorithms, human workers may be used for this task. In a typical crowdsourcing microtask platform, a task manager creates simple tasks for the workers to complete, and the results are combined to produce the final result. Due to the low pay of workers and the difficulty of tasks, individual results may be unreliable.

As an example, consider the task of classifying a dog image into one of four breeds: Pekingese, Mastiff, Maltese, or Saluki. Since workers may not be canine experts, however, they may not be able to directly classify and so we should ask simpler questions. For example, the binary question of whether a dog has a snub nose or a long nose differentiates between {Pekingese, Mastiff} and {Maltese, Saluki}, whereas the binary question of whether the dog is small or large differentiates between {Pekingese, Maltese} and {Mastiff, Saluki}. Using code matrices, we now show how to design binary questions for crowd workers that allow the task manager to reliably infer correct classification even with unreliable workers.

Let the task presented to the crowd be a classification task with M equiprobable classes represented by the hypotheses $H_0, H_1, \ldots, H_{M-1}$. Let N be the number of workers taking part in the task. As part of modeling, let us suppose worker j decides the true class (local decision y_j) with probability p_j and makes the wrong local classification with uniform probability:

$$p(y_j|H_l) = \begin{cases} p_j & \text{if } y_j = l \\ \frac{1-p_j}{M-1} & \text{otherwise.} \end{cases} \quad (1)$$

Let $A = \{a_{lj}\}$ be a binary $M \times N$ code matrix used to design binary questions. For every worker j, let a_j be the corresponding column of A. Each hypothesis $H_l \in \{H_0, H_1, \cdots, H_{M-1}\}$ is associated with a row in the code matrix A. The local workers send a binary answer u_j based on the decision y_j and the corresponding column of code matrix A.

This work was supported in part by ARO under Grant W911NF-09-1-0244, AFOSR under Grants FA9550-10-1-0458 and FA9550-10-1-0263.

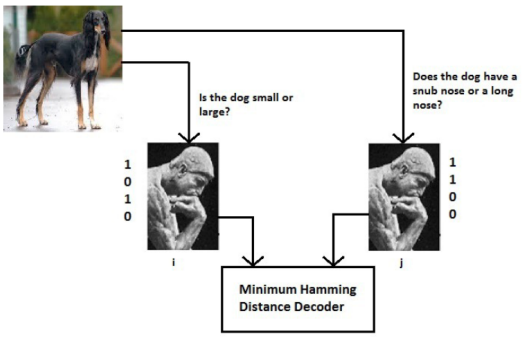

Fig. 1. A schematic diagram showing binary questions posed to workers and the decoding rule used by the task manager.

An illustrative example is shown in Fig. 1 for the dog breed classification task above. Let the columns corresponding to the ith and jth workers be $a_i = [1010]'$ and $a_j = [1100]'$ respectively. The ith worker is asked: "Is the dog small or large?" since she is to differentiate between the first (Pekingese) or third (Maltese) breed and the others. The jth worker is asked: "Does the dog have a snub nose or a long nose?" since she is to differentiate between the first two breeds (Pekingese, Mastiff) and the others. The task manager makes the final classification decision as the hypothesis corresponding to the code word (row) that is closest in Hamming distance to the received vector of decisions. Good codes may be designed using simulated annealing or cyclic column replacement, cf. [15]. The optimality criterion for the code design is to minimize the misclassification probability which is a function of the code matrix as shown later in Sec. 3.

2.1. Unreliable Workers

Although distributed classification in sensor networks and in crowdsourcing are structurally similar, an important difference is the anonymity of crowds. Since the crowd is anonymous, we cannot identify the specific reliability of a specific worker as could potentially be done with a sensor. Hence we assume that each worker j in the crowd has an associated reliability p_j, drawn i.i.d. from a common distribution that characterizes the crowd.

Two crowd reliability models are considered herein. In a spammer-hammer model, the crowd consists of two kinds of workers: spammers and hammers. Spammers are unreliable workers that make a decision at random ($p_j = 1/M$) whereas hammers are reliable workers that make a decision with high reliability. The quality of the crowd, Q, is governed by the fraction of hammers. In a Beta model, the reliabilities of workers are drawn from a Beta distribution with parameters α and β.

3. CLASSIFICATION PERFORMANCE

Having defined a coding approach to reliable crowdsourcing, we determine its performance in terms of average error probability for classification. A traditional approach in crowdsourcing has been to use a majority vote to combine local decisions. In order to compare our coding approach to the majority approach, we also derive its classification performance.

For M-ary classification, each worker's local decision is modeled as $\log_2 M$-bit valued, but since workers only answer binary questions, the N workers are split into $\log_2 M$ groups with each group sending information regarding a single bit. The task manager uses a majority rule to decide each of the $\log_2 M$ bits separately and then concatenates to make the final classification.

Suppose N workers take part in an M-ary classification task. Let $\underline{p}$ denote the reliabilities of these workers, such that p_k for $k = 1, \ldots, N$ are i.i.d. random variables with mean μ. We define this to be an (N, M, μ) crowdsourcing system.

3.1. Error Probability for General Codes

Consider a system with minimum Hamming distance decoding.

Proposition 1. *Consider an (N, M, μ) crowdsourcing system. The expected probability of error using code matrix $A = \{a_{lj}\}$ is:*

$$P_e(\mu) = \frac{1}{M} \sum_{\underline{i},l} \prod_{j=1}^{N} \left[\left(\mu a_{lj} + \frac{(1-\mu)}{(M-1)} \sum_{k \neq l} a_{kj} \right) (2i_j - 1) + (1 - i_j) \right] C^l_{\underline{i}} \quad (2)$$

where $\underline{i} = [i_1, \cdots, i_N] \in \{0,1\}^N$ is the received codeword and $C^l_{\underline{i}}$ is the cost associated with a global decision H_l when the received vector is $\underline{i}$. This cost is:

$$C^l_{\underline{i}} = \begin{cases} 1 - \frac{1}{r} & \text{if } \underline{i} \text{ is in decision region of } H_l \\ 1 & \text{otherwise.} \end{cases} \quad (3)$$

where r is the number of decision regions of $\underline{i}$; r can be greater than one when there is a tie at the task-manager and the tie-breaking rule is to choose one of them randomly.

Proof. Let $P_{e,\underline{p}}$ denote the probability of error given the reliabilities of the N workers. Then, if u_j denotes the bit sent by the worker j and the global decision is made using the Hamming distance criterion,

$$P_{e,\underline{p}} = \frac{1}{M} \sum_{\underline{i},l} P(\underline{u} = \underline{i}|H_l) C^l_{\underline{i}}. \quad (4)$$

Since local decisions are conditionally independent, $P(\underline{u} = \underline{i}|H_l) = \prod_{j=1}^{N} P(u_j = i_j|H_l)$. Further,

$$\begin{aligned} P(u_j = i_j|H_l) &= i_j P(u_j = 1|H_l) + (1 - i_j) P(u_j = 0|H_l) \\ &= (1 - i_j) + (2i_j - 1) P(u_j = 1|H_l) \\ &= (1 - i_j) + (2i_j - 1) \sum_{k=1}^{M} a_{kj} P(y_j = k|H_l) \\ &= (1 - i_j) + \left(p_j a_{lj} + \frac{(1-p_j)}{(M-1)} \sum_{k \neq l} a_{kj} \right) (2i_j - 1) \end{aligned}$$

where y_j is the local decision made by worker j. Since reliabilities p_j are i.i.d. with mean μ, the desired result follows. □

3.2. Error Probability for Majority Voting

Now consider the majority rule, with N divisible by $\log_2 M$.

Proposition 2. *Consider an (N, M, μ) crowdsourcing system. The expected probability of error using majority rule is:*

$$P_e(\mu) = 1 - \frac{1}{M} \left[1 + S_{\tilde{N},(1-q)} \left(\frac{\tilde{N}}{2} \right) - S_{\tilde{N},q} \left(\frac{\tilde{N}}{2} \right) \right]^{\log_2 M} \quad (5)$$

where $\tilde{N} = \frac{N}{\log_2 M}$, $q = \frac{M(1-\mu)}{2(M-1)}$, *and* $S_{N,p}(\cdot)$ *is the survival function (complementary cumulative distribution function) of binomial random variable* $B(N,p)$.

Proof. In a majority approach, $\tilde{N} = \frac{N}{\log_2 M}$ workers send information regarding the ith bit of their local decision, $i = 1, \ldots, \log_2 M$. For a correct global decision, all bits have to be correct. Consider the ith bit and let $P^i_{c,\underline{p}}$ be the probability of the ith bit being correct given the reliabilities of the $\tilde{N}$ workers sending this bit. Then,

$$P^i_{c,\underline{p}} = \frac{P_d + 1 - P_f}{2}, \tag{6}$$

where P_d is the probability of detecting the ith bit as '1' when the true bit is '1' and P_f is the probability of detecting the ith bit as '1' when the true bit is '0'. Note '0' and '1' are equiprobable since all hypotheses are equiprobable. Under majority rule for this ith bit,

$$P_d = \sum_{j=\lfloor \frac{\tilde{N}}{2}+1 \rfloor}^{\tilde{N}} \sum_{\forall G_j} \prod_{k \in G_j} \left(1 - \frac{M(1-p_k)}{2(M-1)}\right) \prod_{k \notin G_j} \frac{M(1-p_k)}{2(M-1)}$$

where G_j is a set of j out of $\tilde{N}$ workers and $\frac{M(1-p_k)}{2(M-1)}$ is the probability of the kth worker making a wrong decision for the ith bit. Similarly,

$$P_f = \sum_{j=\lfloor \frac{\tilde{N}}{2}+1 \rfloor}^{\tilde{N}} \sum_{\forall G_j} \prod_{k \in G_j} \frac{M(1-p_k)}{2(M-1)} \prod_{k \notin G_j} \left(1 - \frac{M(1-p_k)}{2(M-1)}\right)$$

Now the overall probability of correct decision is given by $P_{c,\underline{p}} = \prod_{i=1}^{\log_2 M} P^i_{c,\underline{p}}$. Since reliabilities are i.i.d., the expected probability of correct decision P_c is:

$$P_c = \prod_{i=1}^{\log_2 M} E[P^i_{c,\underline{p}}] \tag{7}$$

where expectation is with respect to $\underline{p}$. Since reliabilities are i.i.d.:

$$E[P_d] = \sum_{j=\lfloor \frac{\tilde{N}}{2}+1 \rfloor}^{\tilde{N}} \binom{\tilde{N}}{j} (1-q)^j q^{(\tilde{N}-j)} = S_{\tilde{N},(1-q)}\left(\frac{\tilde{N}}{2}\right), \tag{8}$$

$$E[P_f] = \sum_{j=\lfloor \frac{\tilde{N}}{2}+1 \rfloor}^{\tilde{N}} \binom{\tilde{N}}{j} q^j (1-q)^{(\tilde{N}-j)} = S_{\tilde{N},q}\left(\frac{\tilde{N}}{2}\right). \tag{9}$$

Using (6), (7), (8), and (9), we get the desired result. □

4. SYSTEM CHARACTERIZATION

We can define an ordering principle for quality of crowds in terms of the quality of their distributed inference performance.

Theorem 1. [ORDERING OF CROWDS] *Consider crowdsourcing systems involving crowd* $\mathcal{C}(\mu)$ *of workers with i.i.d. reliabilities with mean* μ. *Crowd* $\mathcal{C}(\mu)$ *performs better than crowd* $\mathcal{C}(\mu')$ *for inference if and only if* $\mu > \mu'$.

Proof. Follows since average error probabilities depend only on the mean of the reliabilities of the crowd. □

Since the performance criterion is average error probability, this can be regarded as a weak criterion of crowd-ordering in the mean sense. Thus, with this crowd-ordering, better crowds yield better performance in terms of average error probability.

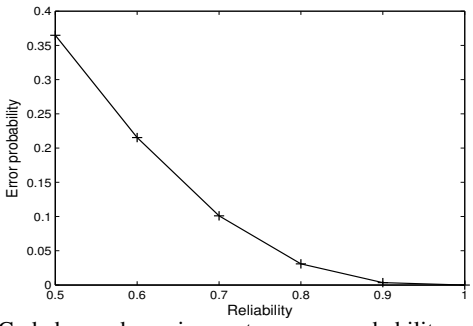

Fig. 2. Coded crowdsourcing system error probability as a function of worker reliability.

4.1. Error probability decreases with more reliable crowds.

Proposition 3. *Average error probability reduces with increasing quality of the crowd.*

Proof. Follows from expressions (2) and (5). □

To get more insight, we simulate a crowdsourcing system with coding as follows: $N = 10$ workers take part in a classification task with $M = 4$ equiprobable classes. A good code matrix C is found by simulated annealing [15]:

$$C = [5, 12, 3, 10, 12, 9, 9, 10, 9, 12]. \tag{10}$$

Here and in the sequel, we represent code matrices as a vector of M bit integers. Each integer r_j represents a column of the code matrix C and can be expressed as $r_j = \sum_{l=0}^{M-1} c_{lj} \times 2^l$. For example, the integer 5 in column 1 of C represents $c_{01} = 1$, $c_{11} = 0$, $c_{21} = 1$ and $c_{31} = 0$.

Let us look at the setting where all workers have the same reliability $p_i = p$. Fig. 2 shows the probability of misclassification as a function of p. As is apparent, probability of misclassification reduces with reliability and approaches 0 as $p \to 1$, as expected.

4.2. Coding is better than majority vote

Now we compare the performance of the coding approach to the majority approach. Fig. 3 shows misclassification probability as a function of crowd quality for $N = 10$ workers taking part in an $(M = 4)$-ary classification task. The spammer-hammer model, where spammers have reliability $p = 1/M$ and hammers have reliability $p = 1$ is used. The figure shows a slight improvement in performance over majority vote when code (10) is used.

We consider a larger system with increased M and N. A good code matrix C for $N = 15$ and $M = 8$ is found by cyclic column replacement:

$$C = [150, 150, 90, 240, 240, 153, 102, 204, 204, 204, 170, 170, 170, 170, 170] \tag{11}$$

The code matrix for the system with $N = 90$ and $M = 8$ is formed sub-optimally by concatenating the columns of (11) six times. Fig. 4 shows the performance when $M = 8$ and N takes the two values: $N = 15$ and $N = 90$. These figures suggest that the gap in performance generally increases for larger system size. Similar observations hold for the Beta model of crowds, see Figs. 5 and 6. A rigorous study on the effect of system size on classification performance is left for future work. Good codes perform better than majority vote as they diversify the binary questions which are asked to

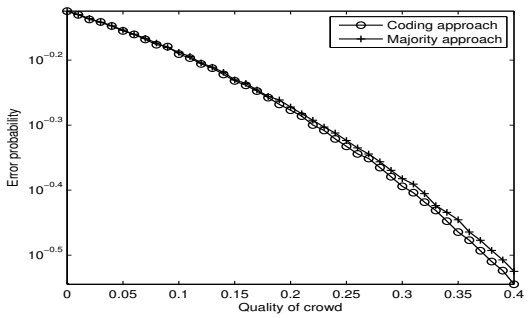

Fig. 3. Error probability as a function of crowd quality using coding and majority approaches with the spammer-hammer model, $(M = 4, N = 10)$.

the workers. From extensive simulation results, we have found that the coding approach is not very sensitive to code matrix C as long as we have approximately equal number of ones and zeroes in every column. However, if we use any code randomly, performance degrades substantially.

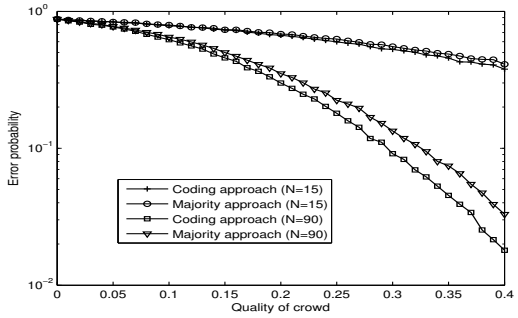

Fig. 4. Error probability as a function of crowd quality using coding and majority approaches with the spammer-hammer model, $(M = 8)$.

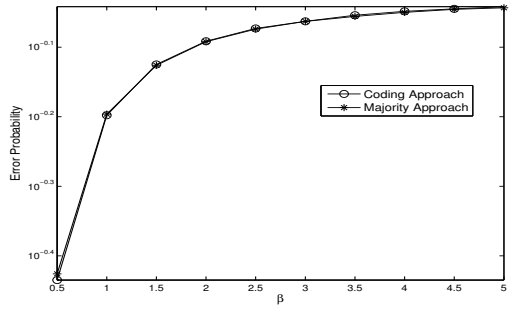

Fig. 5. Error probability as a function of β using coding and majority approaches with the Beta($\alpha = 0.5$, β) model, $(M = 4, N = 10)$.

5. EXPERIMENTAL RESULTS

In this section, we test the proposed coding approach on six publicly available Amazon Mechanical Turk data sets—quantized versions of the data sets in [17]: the anger, disgust, fear, joy, sadness and surprise datasets of the affective text task. Each of the data sets consist of 100 tasks with $N = 10$ workers taking part in each of the tasks.

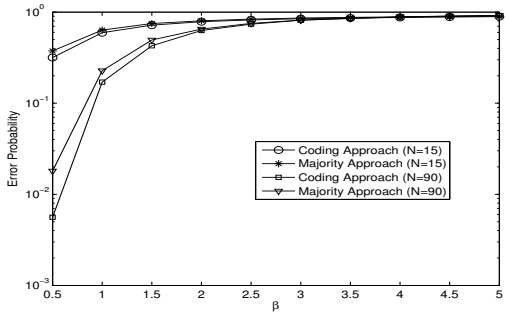

Fig. 6. Error probability as a function of β using coding and majority approaches with the Beta($\alpha = 0.5$, β) model, $(M = 8)$.

Each worker reports a value between 0 and 100, and there is a gold-standard value for each task. For our analysis, we quantize values by dividing the range into $M = 8$ equal intervals. We compare the majority approach with our proposed coding approach. A good optimal code matrix for $N = 10$ and $M = 8$ is designed by simulated annealing [15]:

$$C = [113, 139, 226, 77, 172, 74, 216, 30, 122] \qquad (12)$$

Table 1 compares the performance of the coding and majority approaches. The values in Table 1 are the fraction of wrong decisions made, as compared with the gold-standard value. As the table indicates, the coding based approach outperforms the majority approach in 4 out of the 6 cases considered. We expect the gap in performance to increase as problem size M and crowd size N increase.

Dataset	Coding approach	Majority approach
Anger	0.31	0.31
Disgust	0.26	0.20
Fear	0.32	0.30
Joy	0.45	0.47
Sadness	0.37	0.39
Surprise	0.59	0.63

Table 1. Fraction of errors using coding and majority approaches

6. CONCLUSION

We have proposed the use of coding for reliable classification using unreliable crowd workers and shown that coding in crowdsourcing can more efficiently use human cognitive energy over traditional majority-vote methods. Since minimum Hamming distance decoding is equivalent to MAP decoding in this setting, the anonymity of unreliable crowd workers is not problematic. The benefits of coding are especially large for applications where the number of classes is large, such as fine-grained image classification for building encyclopedias like Visipedia[1]. There one might need to classify among more than 161 breeds of dogs or 10000 species of birds.

Going forward, many further questions may be addressed; two examples are as follows. Can better cognitive and attentional models of human crowd workers provide better insight and design principles? When considering average error probability, the ordering of crowd quality depends only on a first-moment characterization; what about finer characterizations of system performance?

[1]http://www.vision.caltech.edu/visipedia/

7. REFERENCES

[1] D. Bollier, *The Future of Work: What It Means for Individuals, Businesses, Markets and Governments*. Washington, DC: The Aspen Institute, 2011.

[2] D. Tapscott and A. D. Williams, *Wikinomics: How Mass Collaboration Changes Everything*, expanded ed. New York: Portfolio Penguin, 2006.

[3] ——, *Macrowikinomics: Rebooting Business and the World*. New York: Portfolio Penguin, 2010.

[4] J. Marschak and R. Radner, *Economic Theory of Teams*. New Haven: Yale University Press, 1972.

[5] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on Amazon Mechanical Turk," in *Proc. ACM SIGKDD Workshop Human Comput. (HCOMP'10)*, Jul. 2010, pp. 64–67.

[6] M. Lease, "On quality control and machine learning in crowdsourcing," in *Proc. AAAI Workshop Human Comput. (HCOMP'11)*, Aug. 2011, pp. 97–102.

[7] L. R. Varshney, "Privacy and reliability in crowdsourcing service delivery," in *Proc. SRII Global Conf. 2012*, Jul. 2012, pp. 55–60.

[8] P. K. Varshney, *Distributed Detection and Data Fusion*. New York: Springer-Verlag, 1997.

[9] J. Ross, L. Irani, M. S. Silberman, A. Zaldivar, and B. Tomlinson, "Who are the crowdworkers? shifting demographics in Mechanical Turk," in *Proc. 28th SIGCHI Conf. Hum. Factors Comput. Syst. (CHI 2010)*, Apr. 2010, pp. 2863–2872.

[10] J. S. Downs, M. B. Holbrook, S. Sheng, and L. F. Cranor, "Are your participants gaming the system? screening Mechanical Turk workers," in *Proc. 28th SIGCHI Conf. Hum. Factors Comput. Syst. (CHI 2010)*, Apr. 2010, pp. 2399–2402.

[11] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie, "Visual recognition with humans in the loop," in *Computer Vision – ECCV 2010*, ser. Lecture Notes in Computer Science, L. Buttyán, V. Gligor, and D. Westhoff, Eds. Berlin: Springer, 2010, vol. 6314, pp. 438–451.

[12] D. A. Grier, "Error identification and correction in human computation: Lessons from the WPA," in *Proc. AAAI Workshop Human Comput. (HCOMP'11)*, Aug. 2011, pp. 32–36.

[13] D. R. Karger, S. Oh, and D. Shah, "Budget-optimal crowdsourcing using low-rank matrix approximations," in *Proc. 49th Annu. Allerton Conf. Commun. Control Comput.*, Sep. 2011, pp. 284–291.

[14] ——, "Iterative learning for reliable crowdsourcing systems," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds. Cambridge, MA: MIT Press, 2011, pp. 1953–1961.

[15] T.-Y. Wang, Y. S. Han, P. K. Varshney, and P.-N. Chen, "Distributed fault-tolerant classification in wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 724–734, Apr. 2005.

[16] C. Yao, P.-N. Chen, T.-Y. Wang, Y. S. Han, and P. K. Varshney, "Performance analysis and code design for minimum Hamming distance fusion in wireless sensor networks," *IEEE Trans. Inf. Theory*, vol. 53, no. 5, pp. 1716–1734, May 2007.

[17] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng, "Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks," in *Proc. Conf. Empirical Meth. Natural Language Process. (EMNLP'08)*, Oct. 2008, pp. 254–263.