WoCMan: Harnessing the Wisdom of the Crowds for High-Quality Estimates

Daniel W. Barowy Emery D. Berger
University of Massachusetts Amherst
{dbarowy,emery}@cs.umass.edu

Daniel Goldstein Siddharth Suri
Microsoft Research
{dgg,suri}@microsoft.com

1. Problem and Motivation

Estimation is common to many computational problems. “Where are the person’s eyes in this photo?”, “At what time in this audio recording does the interviewee accidentally swear?”, and “How many calories are in the food shown in this image?” are questions where the answer is an estimate of an unknown real value. Estimates are fundamentally approximate. Machine learning-based techniques are capable of producing some estimates, but developing and using such software typically requires expert domain knowledge. Surprisingly, non-expert groups of *people* are also capable of producing accurate estimates. This phenomenon, known as the *wisdom of the crowds*, holds promise in making estimation tasks accessible to ordinary programmers.

We introduce WoCMan, a domain-specific language (DSL) designed to make it easy for programmers to obtain high-quality estimates from the crowd. WoCMan obtains *interval estimates* over arbitrary user-defined functions of crowd responses. Programmers declare their desired *precision* and *budget*, and WoCMan iteratively increases the sample size until either the estimate is sufficiently refined or the budget is exhausted. We demonstrate with a “calorie counting camera” app.

Figure 1. One of 208 school lunch images labeled with ground truth nutritional data.

2. Background and Related Work

Machine Learning. Techniques from machine learning are in some cases capable of answering estimation queries, but developing and using such software typically requires expert knowledge [2, 15]. The difficulty is compounded by the fact that most algorithms require ground truth training data [8]. Such data is frequently obtained via crowdsourcing.

The Wisdom of the Crowds. Crowdsourcing suggests a promising approach. Galton noted that “the middlemost value” of a large number of estimates is often a better estimate of the true value than any individual’s judgement, even when respondents are experts [7]. Estimation theory provides a principled basis for aggregating responses [14], but requires competence in statistics. Crowdsourcing compounds the difficulty since programmers must pay workers, address low-quality or wrong responses, and timeouts [12].

Prior work incorporates crowdsourcing into ordinary programs in a variety of ways. While some have built-in quality control mechanisms, none address quality control for continuous random variables [3–5, 9–11].

Contributions. WoCMan is the first crowdsourcing language to address quality for estimation tasks. WoCMan significantly extends our prior work on AutoMan, a DSL that abstracts crowdsourcing as ordinary function calls [3]. WoCMan augments AutoMan to handle *continuous random variables*. By default, AutoMan provides quality control for *discrete random variables*. The difference is how crowd consensus is reached. Intuitively, AutoMan’s quality control requires agreement on a *particular* answer (e.g., “Does this picture contain a giraffe?”) whereas WoCMan requires only that answers are in the same ballpark (e.g., “How much does this ox weigh in lbs?”). WoCMan inherits AutoMan’s automatic pricing, scheduling, and i.i.d. sampling guarantees.

3. Approach and Uniqueness

Programmers specify estimation tasks declaratively in Scala. The following shows a task specification that makes a task callable as an ordinary function. Note that the following specifies a query, a budget, a (symmetric) confidence interval (CI) width, and with the default confidence level of 0.95.

```scala
def numCalories(url: String) = Estimate (
  budget = 5.00,
  confidence_interval = SymmetricCI(100),
  text = "How many calories are in the food
       pictured?",
  image_url = url
)
```

The goal of WoCMan is to estimate an unknown *parameter* θ of an unknown distribution F of crowd responses on space $\mathcal{X}$. Let $X = (x_1, \ldots, x_n)$ be a real-valued, i.i.d. sample of responses from F of size n.

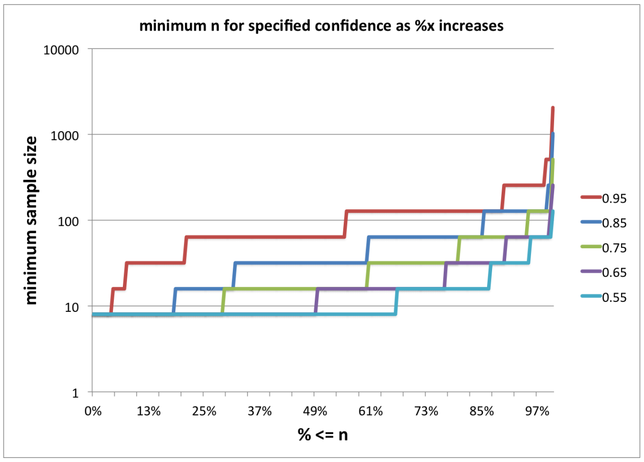

Figure 2. The number of workers (sample size) required by $x\%$ of tasks to reach confidence. Each line represents a distinct confidence setting. Higher confidence settings require more workers. Note: y-axis is log-scale.

Point Estimation. Let $\hat{\theta}(X)$ be an arbitrary real-valued *statistic* (a function of X), the *point estimate of* θ.

Interval Estimation. The fact that $\hat{\theta}$ is an arbitrary, user-defined function over an unknown distribution F complicates interval estimation since precise confidence bounds are only known for specific statistics (e.g., the mean) of known distributions (e.g., the normal distribution). Nonetheless, so-called *non-parametric* methods, which relax parametric assumptions, can be used to estimate arbitrary $\hat{\theta}$ with surprising accuracy [16].

We use the *basic bootstrap* procedure to estimate parameters [1]. The bootstrap produces an estimate of parameter θ, denoted $\hat{\theta}^*$, by way of the estimator $\hat{\theta}$ and random replicates of X that we denote X^*. Let $\hat{F}$ be the empirical distribution such that each $x \in X$ contributes $1/n$ mass. Let B be the number of *bootstrap replications*. For each b from $1 \ldots B$, a bootstrap sample X_b^* is drawn from $\hat{F}$ randomly with replacement and used to compute the bth *bootstrap replication* $\hat{\theta}^*(b)$. For many parameters, bootstrap estimates converge quickly, and in the presence of small deviations from parametric assumptions, are often more accurate and converge faster [6].

The *percentile method* is used to calculate the CI [6]. Let $\hat{\theta}(\alpha) = \widehat{CDF}^{-1}(\alpha)$, a function that returns the *real value* corresponding to the $(1-\alpha) \cdot 100$th percentile of bootstrap replicates $\hat{\theta}^*$. Thus $\theta \in [\hat{\theta}(\alpha), \hat{\theta}(1-\alpha)]$. As $n \to \infty$, $[\hat{\theta}(\alpha), \hat{\theta}(1-\alpha)]$ will include θ with probability $1-\alpha$.

Sample Size Determination. WoCMan's default sample size is 8. There are two outcomes after sampling: 1) the CI satisfies the user's width and confidence level, or 2) it does not. If 1), WoCMan returns the estimate and CI. Otherwise, it refines by obtaining another sample from the crowd. WoCMan doubles the sample size after each iteration.

The bound estimated by WoCMan may accurately reflect the true variability of the population but not meet user constraints. Thus the budget parameter serves as a limiting factor on the total sample size, ensuring that estimation always terminates at a reasonable cost.

4. Preliminary Results

We evaluated WoCMan using a data set of 208 school lunch photos paired with ground truth nutritional data (Fig. 1). WoCMan was run with a fixed CI width of 100 calories, we varied our confidence parameter between 0.55 and 0.95, and measured the number of responses required to satisfy user constraints (See Fig. 2). We ran a second experiment with a fixed confidence (0.95) and varied CI widths between 100 and 500 (not shown).

WoCMan automatically recruits more workers to meet tighter constraints on confidence thresholds. WoCMan needed an average of 111.4 responses for the highest confidence threshold (0.95) vs 14.9 for the lowest (0.55). Likewise, when CI widths are narrowed, WoCMan recruits more workers. WoCMan needed an average of 107.3 responses for the tightest CI (width = 100; mean cost: \$2.14; median cost: \$1.28) vs 9 for the widest (width = 500; mean cost: \$0.47; median cost: \$0.32).

WoCMan compares favorably against the state of the art vision-based solution from Google, IM2Calories [13]. IM2Calories' best performing algorithm had a mean absolute error (MAE) of 152.95 kcal with a standard error (SE) of 15.61 kcal. WoCMan's best performing setting ($1-\alpha = 0.95$; CI width $= 100$) had a MAE of 103.08 kcal with an SE of 6.00. While WoCMan is more expensive than IM2Calories, both in terms of latency and cost, the equivalent WoCMan program (shown above) is trivial to write. IM2Calories also requires that a user's GPS be active so that the appropriate restaurant menu can be located and searched. WoCMan has no such restriction.

References

[1] ISSN 08834237. URL http://www.jstor.org/stable/2246110.

[2] Y. Baba and H. Kashima. Statistical quality estimation for general crowdsourcing tasks. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 554–562, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2174-7. . URL http://doi.acm.org/10.1145/2487575.2487600.

[3] D. W. Barowy, C. Curtsinger, E. D. Berger, and A. McGregor. AutoMan: a platform for integrating human-based and digital computation. In *OOPSLA 2012*, pages 639–654. ISBN 978-1-4503-1561-6. . URL http://doi.acm.org/10.1145/2384616.2384663.

[4] J. Bornholt, T. Mytkowicz, and K. S. Mckinley. Uncertain<T >: A first-order type for uncertain data. In *In ASPLOS*, 2014.

[5] P. Dai, M. Daniel, and S. Weld. Artificial intelligence for artificial artificial intelligence. In *In Proceedings of the 25th AAAI Conference on Artificial Intelligence; AAAI*, pages 1153–1159. Press, 2011.

[6] B. Efron. Nonparametric standard errors and confidence intervals. *Canadian Journal of Statistics*, 9(2):139–158, 1981. ISSN 1708-945X. . URL http://dx.doi.org/10.2307/3314608.

[7] F. Galton. Vox Populi. *Nature*, 75(1949):450–451, Mar. 1907. URL http://www.nature.com/doifinder/10.1038/075450a0.

[8] Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng. Building high-level features using large scale unsupervised learning. In *International Conference in Machine Learning*, 2012.

[9] C. H. Lin, M. Daniel, and S. Weld. Crowdsourcing control: Moving beyond multiple choice.

[10] G. Little, L. B. Chilton, M. Goldman, and R. C. Miller. TurKit: Human Computation Algorithms on Mechanical Turk. In *UIST 2010*, pages 57–66. .

[11] B. Livshits and T. Mytkowicz. Saving money while polling with interpoll using power analysis, 2014.

[12] W. Mason and S. Suri. Conducting Behavioral Research on Amazon's Mechanical Turk. *Social Science Research Network Working Paper Series*, Oct. 2010. URL http://ssrn.com/abstract=1691163.

[13] A. Myers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. Murphy. Im2calories: towards an automated mobile vision food diary. In *ICCV*, 2015.

[14] J. Neyman. Outline of a theory of statistical estimation based on the classical theory of probability. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 236(767):333–380, 1937. ISSN 0080-4614. .

[15] S. Sonnenburg, M. L. Braun, C. S. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K.-R. Müller, F. Pereira, C. E. Rasmussen, et al. The need for open source software in machine learning. *Made available in DSpace on 2010-12-20T06: 05: 49Z (GMT). No. of bitstreams: 1 Sonnenburg_Need2007. pdf: 1278865 bytes, checksum: 31b77a03c5967cafb7381eee2f47fe56 (MD5) Previous issue date: 2009-05-22T01: 55: 10Z*, 2007.

[16] L. Wasserman. *All of Nonparametric Statistics*. Springer Texts in Statistics. Springer New York, 2006. ISBN 9780387306230. URL `https://books.google.com/books?id=MRFlzQfRg7UC`.