Aggregating information from the crowd

Anirban Dasgupta
IIT Gandhinagar

Joint work with Flavio Chiericetti, Nilesh Dalvi, Vibhor Rastogi, Ravi Kumar, Silvio Lattanzi

January 07, 2015
Crowdsourcing

Many different modes of crowdsourcing
Aggregating information using the Crowd: the expertise issue

Typically, the answers to the crowdsourced tasks are unknown!
Aggregating information using the Crowd: the effort issue

Does this article have appropriate references at all places?

Yes Yes Yes No No

Even expert users need to spend effort to give meaningful answers
Elicitation & Aggregation

• How to ensure that information collected is “useful”?  
  – Assume users are strategic  
  – effort put in when making judgments, truthful opinions  
  – design the right payment mechanism

• How to aggregate opinions from different agents?  
  – user behavior stochastic  
  – varying levels of expertise, unknown  
  – might not stick around to develop reputation
This talk: only aggregation

- Formalizing a simple crowdsourcing task
  - Tasks with hidden labels, varying user expertise
- Aggregation for binary tasks
  - Stochastic model of user behaviour
  - Algorithms to estimate task labels + expertise
- Continuous feedback
- Ranking
Binary Task model

- Tasks have hidden labels:
  - \{-1, +1\}
  - E.g. labeling whether good quality article
- Each task is evaluated by a number of users
  - not too many
- Each user outputs \{-1, +1\} per task
- Users and tasks fixed
Simple User model

[Dawid, Skene, '79]

- Each user performs set of tasks assigned to her
- Users have proficiency $p_i$
  - Indicates probability that the true signal is seen
  - This is not observable

Note: This does not model bias
Stochastic model

G = user-item graph
q = vector of actual qualities
$U_{ji} =$ rating on by user j on item i

$U_{ji} = \begin{cases} 
0 & \text{if } G_{ji} = 0 \\
q_i & \text{w.p. } p_j \\
-q_i & \text{w.p. } 1 - p_j
\end{cases}$

Given n-by-m matrix U, estimate vectors q and p
From users to items

- If all users are same, then simple majority/average will do

- Else, some notion of weighted majority e.g.

\[ \tilde{q} = \sum_{j} U_{ji} w_j \]

- We will try to estimate user reliabilities first

\[ w_j = 2p_j - 1 \]
Intuition: if G is complete

- Consider the user x user matrix $UU^t$

$UU^t = (#\text{agreements} - #\text{disagreements})$ between $j$ and $k$

\[
E[UU^t_{jk}] = m(p_j p_k + (1 - p_j)(1 - p_k) - p_j(1 - p_k) - p_k(1 - p_j))
= m(1 - 2p_j)(1 - 2p_k) = mw_j w_k
\]

$E[UU^t]$ is a rank one matrix

\[
UU^t = E[UU^t] + \text{noise}
\]

If we approximate, $UU^t \approx E(UU^t)$, $w$ is rank-1 approximation of $UU^t$
Arbitrary assignment graphs

Hadamard product:

$$(M \otimes N)_{ij} = M_{ij} N_{ij}$$

Then

$$E[UU^t] \approx GG^t \otimes w w^t$$

Number of shared items

E[agree – disagree] on each
Arbitrary assignment graphs

Hadamard product:

\[(M \otimes N)_{ij} = M_{ij}N_{ij}\]

Then

\[E[UU^t] \approx GG^t \otimes ww^t\]

Similar spectral intuitions hold, only slightly more work is needed.
Algorithms

- Core idea is to recover the “expected” matrix using spectral techniques
- **Ghosh, Kale, McAfee'11**
  - compute topmost eigenvector of item x item matrix
  - proves small error for G dense random graph
- **Karger, Oh, Shah'11**
  - using belief propagation on U
  - proof of convergence for G sparse random
- **Dalvi, D., Kumar, Rastogi'13**
  - for G an “expander”, use eigenvectors of both GG' and UU'
- EM based recovery **Dawid & Skene'79**
Empirical: user proficiency can be more or less estimated

Correlation of predicted and actual proficiency on the Y-axis

[ Aggregating crowdsourced binary ratings, WWW'13
Dalvi, D., Kumar, Rastogi ]
Aggregation

Formalizing a simple crowdsourcing task
  - Tasks with hidden labels, varying user expertise

Aggregation for binary tasks
  - Stochastic model of user behaviour
  - Algorithms to estimate task labels + expertise

Continuous feedback

Ranking
Continuous feedback model

- Tasks are continuous:
  - Quality $\mu_i$
- Each user has a reliability $\sigma_j$
- Each user outputs a score per task

$$\sigma_1 \leq \sigma_2 \leq \ldots \leq \sigma_n$$
Continuous feedback model

- Tasks are continuous:
  - Quality $\mu_i$
- Each user has a reliability $\sigma_j$
- Each user outputs a score per task
  \[ U_{ji} \sim N(\mu_i, \sigma_j) \]

Minimize $\max E[|\mu_i - \hat{\mu}_i|]$
Some simpler settings & obstacles
Single item, known variances

Suppose that we know the $\sigma_i$

We want to minimize $E[|\mu - \hat{\mu}|]$
Single item, known variances

Suppose that we know the $\sigma_i$

We want to minimize

\[ E[|\mu - \hat{\mu}|] \]

\[ = \frac{\sum_{j=1}^{n} \frac{x_j}{\sigma_j^2}}{\sum_{j=1}^{n} \frac{1}{\sigma_j^2}} \]

it is known that an asymptotically optimal estimate is

\[ \text{Loss} = E[|\mu - \hat{\mu}|] = \left( \sum_j \frac{1}{\sigma_j^2} \right)^{-1/2} \]
Single item, *unknown variances*

Suppose that we *do not know* the $\sigma_i$

Only one sample, so cannot estimate $\sigma_i$

Cannot compute weighted average

We want to minimize

$$E[|\mu - \hat{\mu}|]$$
Arithmetic Mean

In binary case for single item we can obtain the optimum by using a majority rule.

In a continuous case using the same approach we would compute the arithmetic mean.

\[ \hat{\mu} = \frac{1}{n} \sum_{j=1}^{n} x_j \]
Arithmetic Mean

In binary case for single item we can obtain the optimum by using a majority rule.

In a continuous case using the same approach we would compute the arithmetic mean

$$\hat{\mu} = \frac{1}{n} \sum_{j=1}^{n} x_j$$

and hence

$$\hat{\mu} \sim N \left( \mu, \frac{\sum_{j=1}^{n} \sigma_j^2}{n^2} \right)$$
Arithmetic Mean

In binary case for single item we can obtain the optimum by using a majority rule.

In a continuous case using the same approach we would compute the arithmetic mean

\[ \hat{\mu} = \frac{1}{n} \sum_{j=1}^{n} x_j \]

and hence

\[ \hat{\mu} \sim N \left( \mu, \frac{\sum_{j=1}^{n} \sigma_j^2}{n^2} \right) \]

Thus the loss

\[ \Theta \left( \frac{\sqrt{\sum_{j=1}^{n} \sigma_j^2}}{n} \right) \]
Arithmetic Mean

In binary case for single item we can obtain the optimum by using a majority rule.

In a continuous case using the same approach we would compute the arithmetic mean

\[
\hat{\mu} = \frac{1}{n} \sum_{j=1}^{n} x_j
\]

and hence

\[
\hat{\mu} \sim N \left( \mu, \frac{\sum_{j=1}^{n} \sigma_j^2}{n^2} \right)
\]

Thus the loss

\[
\Theta \left( \frac{\sqrt{\sum_{j=1}^{n} \sigma_j^2}}{n} \right)
\]

Is this optimal?
Problem with Arithmetic mean

\[ \sigma_1 = 1 \]
\[ \sigma_2 = 1 \]
\[ \sigma_3 = n^5 \]
\[ \sigma_n = n^5 \]

The AM would have error \( \Theta \left( n^{4.5} \right) \)
Problem with Arithmetic mean

$\sigma_1 = 1$

$\sigma_2 = 1$

$\sigma_3 = n^5$

$\sigma_n = n^5$

The AM would have error $\Theta(n^{4.5})$

Same problem with the median algorithm
Problem with Arithmetic mean

\[ \sigma_1 = 1 \]
\[ \sigma_2 = 1 \]
\[ \sigma_3 = n^5 \]
\[ \sigma_n = n^5 \]

The AM would have error \( \Theta \left( n^{4.5} \right) \)

Same problem with the median algorithm

By choosing the nearest pair of points, we have a much better estimate
Shortest gap algorithm

Maybe the optimal algo is to select one of two nearest samples?

\[ \sigma_1 = 1 \]
\[ \sigma_2 = 1 \]
\[ \sigma_3 = 1 \]
\[ \sigma_n = 1 \]

In this setting, w.h.p., the two closest points are at distance \( \Theta(1) \)
But arithmetic mean gives loss \( \Theta(n^{-1/2}) \)
Last obstacle

More is not always better

\[ \sigma_1 = 1 \]
\[ \sigma_2 = 1 \]
\[ \sigma_3 = n^5 \]
\[ \sigma_{n^4} = n^5 \]

Adding bad raters could actually worsen the shortest gap algorithm

Mean is not good here either

In this setting, w.h.p., the first two closest points are at distance \( \Theta(1) \)

But so will be some other pair
Single Item case
Results

$\sigma_1 \leq \sigma_2 \leq \ldots \leq \sigma_n$

Theorem 1: There is an algo with expected loss

$\tilde{O}(\sqrt{n} \cdot \sigma_{\log n})$

Theorem 2: There is an example where the gap between any algo and the known variance setting is

$\tilde{\Omega}(\sqrt{n} \cdot \sigma_{\log n})$

[Chiericetti, D., Kumar, Lattanzi' 14]
Algorithm

Combination of two simple algorithms

\textit{k-median algorithm}

return the rate of one of the \( k \) central raters
Algorithm

Combination of two simple algorithms

k-median algorithm

return the rate of one of the k central raters
Algorithm

Combination of two simple algorithms

**k-median algorithm**
return the rate of one of the k central raters

```
r1  r2  r3  r4  r5  r6  r7  r8  r9  r10
```

**k-shortest gap**
Return one of the k closest points

```
r1  r2  r3  r4  r5  r6  r7  r8  r9  r10
```
Algorithm

Combination of two simple algorithms

- **k-median algorithm**
  - return the rate of one of the k central raters

- **k-shortest gap**
  - Return one of the k closest points
Algorithm

Let $l_k$ be the length of the $k$-shortest gap

Compute the $4\sqrt{cn \log n}$ median
Find the $\log n$ shortest gap and return a point in it
Proof Sketch

WHP \( l_k \), length of the k-shortest gap is at most \( \sigma_k \log n \)

Select the \( 4\sqrt{cn \log n} \) median points

w.h.p. contains \( \mu \)
Proof Sketch

WHP $l_k$, length of the k-shortest gap is at most $\sigma_k \log n$

Select the $4\sqrt{cn \log n}$ median points

If we consider $4\sqrt{cn \log n}$ points, then WHP there will be no $\log n$ ratings with variance than $\sqrt{n} \cdot \sigma_{\log n} \log^3 n$

that are within distance $4\sigma_k \log n$
Proof Sketch

Thus the distance of the $\log(n)$ shortest gap points to the truth is bounded

$$\mu$$  
$$\sigma_i = \sqrt{n} \cdot \sigma_{\log n} \log^3 n$$  
$$\sqrt{n} \cdot \sigma_{\log n} \log^3 n + l_{\log n}$$
Lower bound

**Instance:** $\mu$ selected in $\{-L, +L\}$

\[
\text{variance of } j\text{-th user} = \begin{cases} 
p^2n & \text{with prob. } p \\
(1 - p) & \text{with prob. } 1 - p. 
\end{cases}
\]

\[L = n^{-1/2} \quad p = \frac{\log n}{n}\]

Optimal algorithm (known variance) has loss $O\left(\frac{\log^{1.5} n}{n}\right)$
Lower bound

**Instance:** $\mu$ selected at random in $\{-L, +L\}$

\[
\text{variance of j-th user} = \begin{cases} 
p^2n \text{ with prob. } p \\
(1 - p) \text{ with prob. } 1 - p.
\end{cases}
\]

\[
L = n^{-1/2}, \quad p = \frac{\log n}{n}
\]

Optimal algorithm (known variance) has loss $O\left(\frac{\log^{1.5} n}{n}\right)$

We will show that maximum likelihood estimation cannot distinguish between $-L$ and $+L$ → loss $O(n^{-1/2})$
Lower Bound

Consider the two log-likelihoods

\[ \mathcal{L}_- = (2\pi)^{-\frac{n}{2}} \prod_{i=1}^{n} \left( \frac{p}{p^2 n} e^{-(p^2 n)^{-2}(x_i+L)^2/2} + \frac{1-p}{1-p} e^{-(1-p)^{-2}(x_i+L)^2/2} \right) \]

\[ \mathcal{L}_+ = (2\pi)^{-\frac{n}{2}} \prod_{i=1}^{n} \left( \frac{p}{p^2 n} e^{-(p^2 n)^{-2}(x_i-L)^2/2} + \frac{1-p}{1-p} e^{-(1-p)^{-2}(x_i-L)^2/2} \right) \]

Claim: Irrespective of value of \( \mu \), \( \log \left( \frac{\mathcal{L}_+}{\mathcal{L}_-} \right) \) can be positive or negative with const prob.
Lower Bound

Consider the two log-likelihoods

\[ \mathcal{L}_- = (2\pi)^{-\frac{n}{2}} \prod_{i=1}^{n} \left( \frac{p}{p^2 n} e^{-(p^2 n)^{-2}(x_i + L)^2 / 2} + \frac{1-p}{1-p} e^{-(1-p)^{-2}(x_i + L)^2 / 2} \right) \]

\[ \mathcal{L}_+ = (2\pi)^{-\frac{n}{2}} \prod_{i=1}^{n} \left( \frac{p}{p^2 n} e^{-(p^2 n)^{-2}(x_i - L)^2 / 2} + \frac{1-p}{1-p} e^{-(1-p)^{-2}(x_i - L)^2 / 2} \right) \]

**Claim:** Irrespective of value of \( \mu \), \( \log \left( \frac{\mathcal{L}_+}{\mathcal{L}_-} \right) \) can be positive or negative with const prob.

\[ \log \left( \frac{\mathcal{L}_+}{\mathcal{L}_-} \right) = \frac{2L}{(1-p)^2} \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} \frac{1 + \frac{1}{pn} e^{-(p^{-4} n^{-2} - (1-p)^{-2})(x_i - L)^2 / 2}}{1 + \frac{1}{pn} e^{-(p^{-4} n^{-2} - (1-p)^{-2})(x_i + L)^2 / 2}} \]
Multiple items

The idea is to use the same algorithm of constant number of items, but to use a smarter version of the k shortest gap that looks for k points at distance at most $l_k$ in all the items

\[ r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8, r_9, r_{10} \]

\[ r_2, r_3, r_1, r_4, r_7, r_6, r_5, r_8, r_9, r_{10} \]
The idea is to use the same algorithm of constant number of items, but to use a smarter version of the $k$ shortest gap that looks for $k$ points at distance at most $l_k$ in all the items.
Multiple items

**Theorem**: For $m = o(\log n)$, complete graph, can get an expected loss of $\tilde{O}(n^{1/m} \sigma_{\log n})$

**Theorem**: For $m = \Omega(\log n)$, complete or dense random, expected loss almost identical to the known variance case
Aggregation

Formalizing a simple crowdsourcing task
- Tasks with hidden labels, varying user expertise

Aggregation for binary task
- Stochastic model of user behaviour
- Algorithms to estimate task labels + expertise

Continuous feedback

Ranking
Crowdsourced rankings

Ranking of artists of Modernism movement

Gaudí Casas Nonell...
Gaudí Rusiñol Camarasa...
Casas Llimona Nonell...
Gaudí Mir Blai...
Crowdsourced rankings

How can we aggregate noisy rankings

Ranking of artists of Modernism movement

Gaudí Casas Nonell ...
Gaudí Rusiñol Camarasa ...
Casas Llimona Nonell ...
Gaudí Mir Blai ...
Crowdsourced rankings

How can we aggregate noisy rankings
Mallows Model [Mallows 1957]

There is a hidden permutation $\sigma$ and a scale parameter $\beta$.

A permutation $\pi$ is generated as

$$P(\pi) \propto e^{-\beta \kappa(\sigma, \pi)}$$

$\kappa(\sigma, \pi) =$ Kendall-Tau distance

Braverman, Mossel'09: Finding the MLE for single parameter Mallows
Mallows Model

There is a hidden permutation $\sigma$ and a user specific scale parameter $\beta_i$

$$P(\pi_i) \propto e^{-\beta_i \kappa(\sigma, \pi)}$$
Single item with known parameters

**Theorem:** For $m$ samples, if $\sum_{u \leq m} \min(\beta_u^2, 1) \geq C \log n$ then can recover $\sigma$ WHP.

**Theorem:** If $\sum_{u \leq m} \min(\beta_u^2, 1) \leq c$ then cannot recover $\sigma$

**Algo:** Weighted Borda count, weights = thresholded $\beta$ values

Approximate reconstruction versions of these theorems also hold

[Chiericetti, D, Kumar, Lattanzi, RANDOM'14]
Summary

- Host of interesting problems in crowdsourcing aggregation
  - Specially for structured outputs
- For binary tasks
  - Spectral techniques provide a powerful tool
- For gaussians
  - new aggregation problems even for single item
  - Combination of k-median & k-shortest gap
- For ranking
  - Main technical contribution is calculating the swapping probs
  - aggregation with known parameters is nontrivial
Open questions

• Continuous feedback
  – More natural algorithms for aggregation?
  – Better algorithms for multiple items
  – Instance optimal algorithms?
  – Non-gaussian distributions?
  – Mixture learning with lots of components and single/constant samples per component?

• Ranking
  – Better estimation of Mallows parameters
  – Multiple items, under partial ranking/pairwise preferences?

• More realistic complex model of user?
  – Incorporating user bias?
  – different kind of expertise, not just reliability
Thanks!