The Importance of Being Earnest in Crowdsourcing Systems

Alberto Tarable[1], Alessandro Nordio[1], Emilio Leonardi[1,2], Marco Ajmone Marsan[1,2,3]

[1]CNR-IEIIT

[2]Politecnico di Torino

[3]IMDEA Networks Institute

Cavalese, Jan 14-th, 2015

Introduction

Crowdsourcing systems

- provide a significant new type of work organization paradigm

Introduction

Crowdsourcing systems

- provide a significant new type of work organization paradigm
- solve a wide range of complex problems in a scalable way

Introduction

Crowdsourcing systems

- provide a significant new type of work organization paradigm
- solve a wide range of complex problems in a scalable way
- integrate a large number of human and/or computer efforts

Introduction

The key characteristic of crowdsourcing systems is that:

Introduction

The key characteristic of crowdsourcing systems is that:

- a *requester* structures his problem in a set of *tasks*

Introduction

The key characteristic of crowdsourcing systems is that:

- a *requester* structures his problem in a set of *tasks*
- it assigns every tasks to a set of *workers*

Introduction

The key characteristic of crowdsourcing systems is that:

- a *requester* structures his problem in a set of *tasks*
- it assigns every tasks to a set of *workers*
- workers provide unreliable *answers*, (for simplicity answers are assumed to be binary)

Introduction

The key characteristic of crowdsourcing systems is that:

- a *requester* structures his problem in a set of *tasks*
- it assigns every tasks to a set of *workers*
- workers provide unreliable *answers*, (for simplicity answers are assumed to be binary)
- the correct task *solution* is obtained from answers through a *decision* rule

Assumptions

- T binary tasks whose outcome is represented by i.i.d. uniform random variables (RV's) $\tau_1, \tau_2, \ldots, \tau_T$ over $\{\pm 1\}$, i.e., $\mathbb{P}\{\tau_t = \pm 1\} = \frac{1}{2}$, $t = 1, \ldots, T$

Assumptions

- T binary tasks whose outcome is represented by i.i.d. uniform random variables (RV's) $\tau_1, \tau_2, \ldots, \tau_T$ over $\{\pm 1\}$, i.e., $\mathbb{P}\{\tau_t = \pm 1\} = \frac{1}{2}$, $t = 1, \ldots, T$
- W workers, each one modeled as a binary symmetric channel (BSC); i.e., providing a wrong answer with probability p_{tw} and a correct answer with probability $1 - p_{tw}$

State of the Art Crowdsourcing Systems

Normally

- every *task* is assigned to K randomly chosen workers (uniform assignment)

State of the Art Crowdsourcing Systems

Normally

- every *task* is assigned to K randomly chosen workers (uniform assignment)
- task *decision* is made by adopting a simple majority rule

State of the Art Crowdsourcing Systems

Normally

- every *task* is assigned to K randomly chosen workers (uniform assignment)
- task *decision* is made by adopting a simple majority rule

Better performance can be achieved by designing smarter assignment schemes and decision rules!

Motivations

The performance of crowdsourcing systems has been explored, but

Motivations

The performance of crowdsourcing systems has been explored, but

- most previous works assume no a-priori information about worker reliability (worker reputation - or earnestness) at the scheduler

Motivations

The performance of crowdsourcing systems has been explored, but

- most previous works assume no a-priori information about worker reliability (worker reputation - or earnestness) at the scheduler
- smart decision rules exploit redundancy and correlation in the pattern of answers to infer an a-posteriori reliability estimate for every worker

Motivations

The performance of crowdsourcing systems has been explored, but

- most previous works assume no a-priori information about worker reliability (worker reputation - or earnestness) at the scheduler
- smart decision rules exploit redundancy and correlation in the pattern of answers to infer an a-posteriori reliability estimate for every worker

Understanding the potential impact of a-priori information about worker reliability is extremely important

Paper Goal

We provide the first systematic analysis of the potential benefits deriving from a-priori knowledge about the reputation of workers

Paper Goal

We provide the first systematic analysis of the potential benefits deriving from a-priori knowledge about the reputation of workers

- workers can be grouped into classes, each one composed of workers with similar accuracy and skills

Paper Goal

We provide the first systematic analysis of the potential benefits deriving from a-priori knowledge about the reputation of workers

- workers can be grouped into classes, each one composed of workers with similar accuracy and skills
  - each worker belongs to one of K classes, $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K$
  - each class is characterized, for each task, by an *average* error probability π_{tk}, known to the requester

Paper Goal

We provide the first systematic analysis of the potential benefits deriving from a-priori knowledge about the reputation of workers

- workers can be grouped into classes, each one composed of workers with similar accuracy and skills
  - each worker belongs to one of K classes, $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K$
  - each class is characterized, for each task, by an *average* error probability π_{tk}, known to the requester
- two extreme scenarios are possible:
  - Full Knowledge: the error probability of each worker in $\mathcal{C}_k$ is deterministically equal to π_{tk} for task t (zero variance case)
  - Hammer-Spammer (HS): perfectly reliable and completely unreliable users coexists within the same class (maximum variance case)

Task assignment

- An *allocation* is a set of assignments of tasks to workers; a generic allocation corresponds to a set $\mathcal{G}$ of pairs (t, w) with $t \in \{1, \cdots, T\}$ and $w \in \{1, \cdots, W\}$

Task assignment

- An *allocation* is a set of assignments of tasks to workers; a generic allocation corresponds to a set $\mathcal{G}$ of pairs (t, w) with $t \in \{1, \cdots, T\}$ and $w \in \{1, \cdots, W\}$
- $\mathcal{O}$ is the complete allocation set ($\mathcal{O}$ is the set composed of all the possible $T \cdot W$ pairs (t, w))

Task assignment

- An *allocation* is a set of assignments of tasks to workers; a generic allocation corresponds to a set $\mathcal{G}$ of pairs (t, w) with $t \in \{1, \cdots, T\}$ and $w \in \{1, \cdots, W\}$
- $\mathcal{O}$ is the complete allocation set ($\mathcal{O}$ is the set composed of all the possible $T \cdot W$ pairs (t, w))
- We impose the following constraints:

Task assignment

- An *allocation* is a set of assignments of tasks to workers; a generic allocation corresponds to a set $\mathcal{G}$ of pairs (t, w) with $t \in \{1, \cdots, T\}$ and $w \in \{1, \cdots, W\}$
- $\mathcal{O}$ is the complete allocation set ($\mathcal{O}$ is the set composed of all the possible $T \cdot W$ pairs (t, w))
- We impose the following constraints:
  - a given task t can be assigned at most once to a given worker w

Task assignment

- An *allocation* is a set of assignments of tasks to workers; a generic allocation corresponds to a set $\mathcal{G}$ of pairs (t, w) with $t \in \{1, \cdots, T\}$ and $w \in \{1, \cdots, W\}$
- $\mathcal{O}$ is the complete allocation set ($\mathcal{O}$ is the set composed of all the possible $T \cdot W$ pairs (t, w))
- We impose the following constraints:
  - a given task t can be assigned at most once to a given worker w
  - no more than r_w tasks can be assigned to worker w

Task assignment

- An *allocation* is a set of assignments of tasks to workers; a generic allocation corresponds to a set $\mathcal{G}$ of pairs (t, w) with $t \in \{1, \cdots, T\}$ and $w \in \{1, \cdots, W\}$
- $\mathcal{O}$ is the complete allocation set ($\mathcal{O}$ is the set composed of all the possible $T \cdot W$ pairs (t, w))
- We impose the following constraints:
  - a given task t can be assigned at most once to a given worker w
  - no more than r_w tasks can be assigned to worker w
  - the total number of assignments cannot be larger than C

Greedy task assignment

The task assignment we propose to approximate the optimum behavior is a simple greedy algorithm that starts from an empty assignment ($\mathcal{G}^{(0)} = \emptyset$), and at every iteration i adds to $\mathcal{G}^{(i-1)}$ the individual assignment $(t, w)^{(i)}$, so as to maximize an objective function P():

Greedy task assignment

The task assignment we propose to approximate the optimum behavior is a simple greedy algorithm that starts from an empty assignment ($\mathcal{G}^{(0)} = \emptyset$), and at every iteration i adds to $\mathcal{G}^{(i-1)}$ the individual assignment $(t, w)^{(i)}$, so as to maximize an objective function P():

$$(t, w)^{(i)} = \underset{(t,w) \in \mathcal{O} \setminus \mathcal{G}^{(i-1)}, (\mathcal{G}^{(i-1)} \cup \{(t,w)\}) \in \boldsymbol{\mathcal{F}}}{\arg\max} P(\mathcal{G}^{(i-1)} \cup \{(t, w)\})$$

The algorithm stops when no assignment can be further added to $\mathcal{G}$ without violating the cost constraint C

Greedy task assignment

Several choices are possible for the objective function P():

Greedy task assignment

Several choices are possible for the objective function P():

- $P_1 = 1 - \frac{1}{T} \sum_t P_{e,t}$
- $P_2 = 1 - \max_t P_{e,t}$
- $P_3 = \sum_{t=1}^{T} I(\mathbf{a}_t; \tau_t)$

Decision Rules

- Majority rule: $\hat{\tau}_t(\mathbf{a}_t) = \operatorname{sgn}\left(\sum_w a_{tw}\right)$

Decision Rules

- Majority rule: $\hat{\tau}_t(\mathbf{a}_t) = \mathrm{sgn}\left(\sum_w a_{tw}\right)$
- MAP rule: $\hat{\tau}_t(\mathbf{a}_t) = \mathrm{sgn}\left(\sum_w a_{tw}\sigma_{k(w)}\right)$ with $\sigma_{k(w)} = log\frac{1-\pi_{w,k(w)}}{\pi_{t,k(w)}}$

Decision Rules

- Majority rule: $\hat{\tau}_t(\mathbf{a}_t) = \mathrm{sgn}\left(\sum_w a_{tw}\right)$
- MAP rule: $\hat{\tau}_t(\mathbf{a}_t) = \mathrm{sgn}\left(\sum_w a_{tw}\sigma_{k(w)}\right)$ with $\sigma_{k(w)} = log\frac{1-\pi_{w,k(w)}}{\pi_{t,k(w)}}$
- Low Rank Approximation (LRA) rule [1]:

$$\hat{\tau}_t(\mathbf{a}_t) = \mathrm{sgn}\left(\sum_w a_{tw} v_w\right)$$

where v_w are the components of the leading right singular vector associated with the matrix of answers $[a_{tw}]$

---

[1] D. R. Karger, S. Oh, D. Shah, "Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems," *Operations Research*, vol. 62, no. 1, pp. 1–24, 2014.

Results: Considered Algorithms

- "Majority voting" + "Uniform allocation" $\rightarrow$ "Majority"

Results: Considered Algorithms

- "Majority voting" + "Uniform allocation" $\rightarrow$ "Majority"
- "LRA" + "Uniform allocation" $\rightarrow$ "LRA uniform"

Results: Considered Algorithms

- “Majority voting” + “Uniform allocation” $\rightarrow$ “Majority”
- “LRA” + “Uniform allocation” $\rightarrow$ “LRA uniform”
- “LRA” + “Greedy allocation” $\rightarrow$ “LRA greedy”

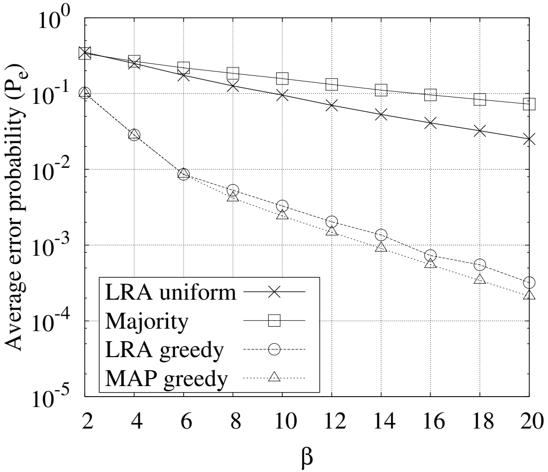
Results: Considered Algorithms

- "Majority voting" + "Uniform allocation" $\rightarrow$ "Majority"
- "LRA" + "Uniform allocation" $\rightarrow$ "LRA uniform"
- "LRA" + "Greedy allocation" $\rightarrow$ "LRA greedy"
- "MAP" + "Greedy allocation" $\rightarrow$ "MAP greedy"

Results: a first scenario

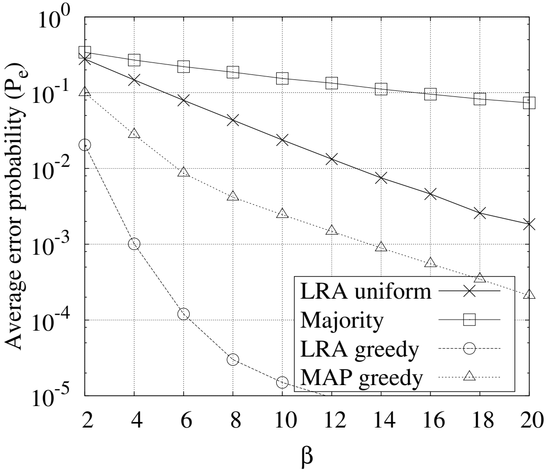
- Number of i.i.d tasks: $T = 100$
- 3 classes of workers: $\pi_{t1} = 0.1, \pi_{t2} = 0.2, \pi_{t3} = 0.5$
- Number of workers per class: $W_1 = 30$, $W_2 = 120$, and $W_3 = 150$
- Maximum number of tasks per worker: $r_w = 20$

Full Knowledge



β is the average number workers per task

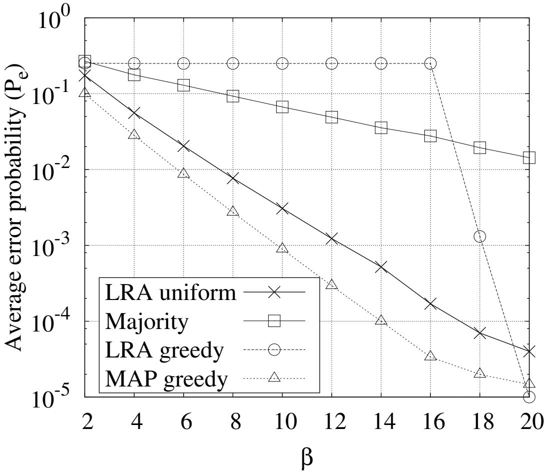
Hammer-Spammer



Results: a second scenario

- Two groups of 50 tasks each
- Error probabilities for the tasks in group 1 and 2 are given by
  $\pi_{t_1 1} = 0.1, \pi_{t_1 2} = 0.25, \pi_{t_1 3} = 0.5$
  $\pi_{t_2 1} = 0.5, \pi_{t_2 2} = 0.25, \pi_{t_2 3} = 0.1$
- Number of workers per class: $W_1 = 40, W_2 = 120$, and $W_3 = 40$
- Maximum number of tasks per worker: $r_w = 20$

Hammer-Spammer



Several other results in the paper!

Conclusions

Main findings of our paper are:

Conclusions

Main findings of our paper are:

- even largely inaccurate estimates of workers' reputation during task assignment $\rightarrow$ large improvements of system performance

Conclusions

Main findings of our paper are:

- even largely inaccurate estimates of workers' reputation during task assignment $\rightarrow$ large improvements of system performance
- a simple optimal task-independent MAP decision rule is proposed for the case of full knowledge of workers' reputation

Conclusions

Main findings of our paper are:

- even largely inaccurate estimates of workers' reputation during task assignment $\rightarrow$ large improvements of system performance
- a simple optimal task-independent MAP decision rule is proposed for the case of full knowledge of workers' reputation
- when workers' reputation estimates are significantly inaccurate, the best performance can be obtained by combining our proposed task assignment algorithm with advanced decision rules such as LRA

Many, many thanks!

Many, many thanks! Questions?