Coding Theory for Reliable Signal Processing

Yunghsiang S. Han(韓永祥)

Department of Electrical Engineering,
National Taiwan University of Science and Technology(國立台灣科技大學)

January 29, 2015

Outline

- Introduction
  - Reliable Signal Processing
  - Coding Theory
- DCFECC Approach
- Distributed *M*-ary Classification
  - Fault-tolerant Distributed Classification
  - Numerical Results
- Secure Target Localization
  - Localization as Hierarchical Classification
  - Numerical Results
- Reliable Crowdsourcing
  - Coding for Crowdsourcing
  - Numerical Results
  - Experimental Results
- Conclusion

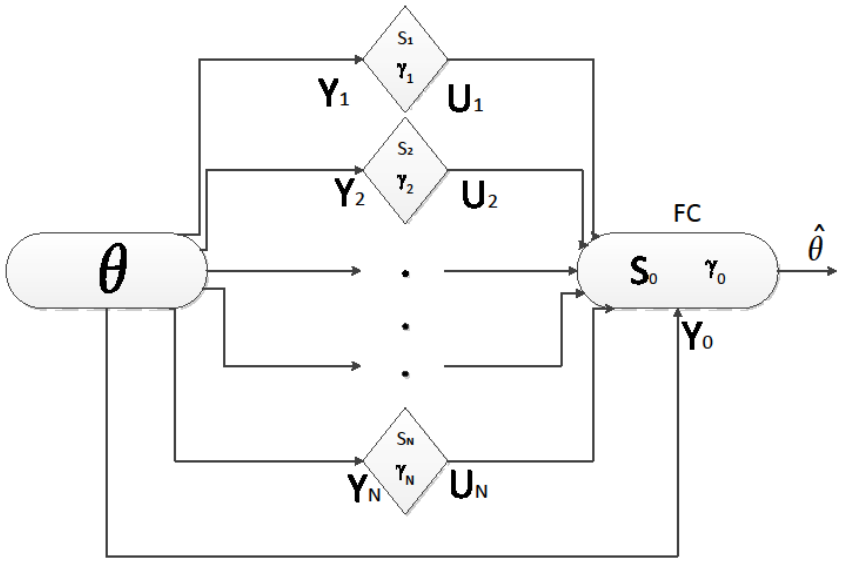
Reliable Signal Processing

- Increased dependence on technology in everyday life
- Need to ensure reliable performance
- Systems can fail due to multiple reasons:
  - presence of a component with permanent failure,
  - a malicious component providing corrupt information, or
  - an unreliable component which randomly provides faulty data.
- Design systems to perform reliably in the presence of such unreliable components.

Coding Theory

- Coding theory: a possible solution
- Used for error correction in data communication and storage
- More recently applied to field of networked data storage systems
- **Focus**: Application to Distributed Inference Networks

Distributed Inference Networks

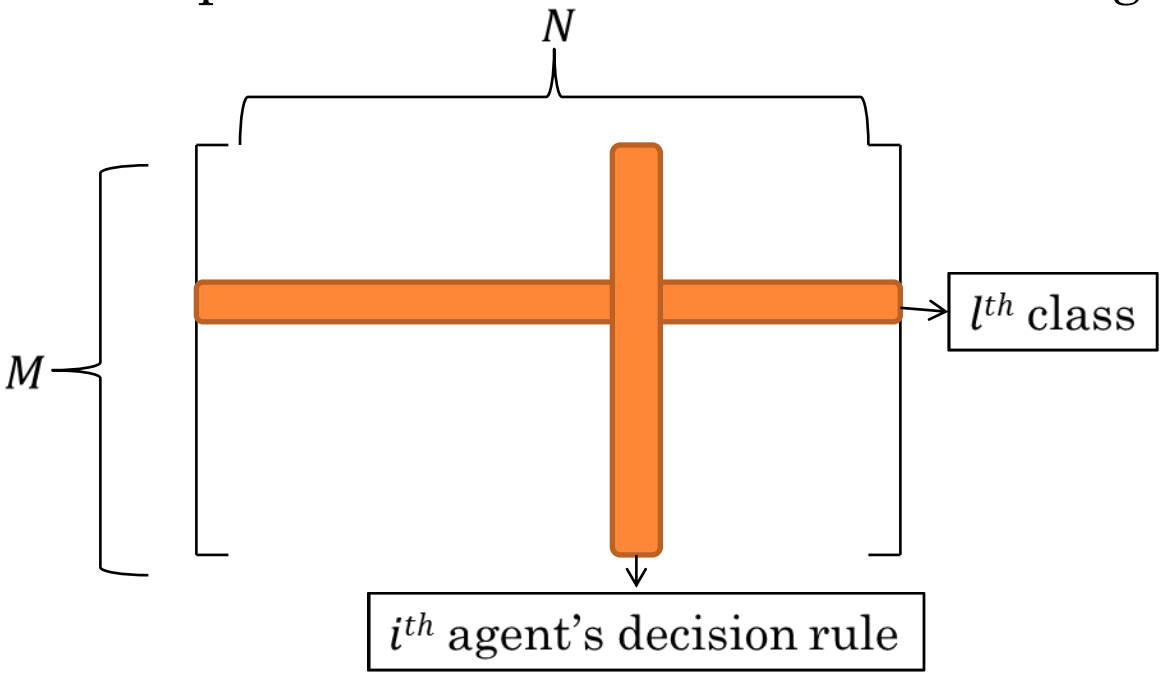


Typical Distributed Inference Network

- Network consisting of local agents make observations
- Send their inference to a central unit called Fusion Center (FC)
- Agents: physical sensors or human decision makers
- FC fuses the data to make a final inference
- Erroneous data from these local agents would result in a degraded performance

DCFECC Approach (Wang *et al.*, 2005)

- Simple idea: Represent the classification problem using a binary code matrix ***C***
- *M* hypotheses and *N* agents: ***C*** is *M* x *N*
- Each row corresponds to one of the different possible hypotheses
- Columns represent the decision rules of the agents



DCFECC Approach (contd..)

- Agent i sends its binary decision ($u_i \in \{0,1\}$) using the quantization rule corresponding to the i^{th} column.
- For example, if agent i decides hypothesis H_j, it sends binary bit-value corresponding to $(j, i)^{th}$ element of C, i.e., $u_i = c_{ji}$.
- FC receives the N-bit vector, $u = [u_1, \cdots, u_N]$
- Final classification decision using minimum Hamming distance based fusion

Implications of DCFECC

- Error-correction property of the code matrix provides the fault-tolerance capability
- Code matrix used for local decision rules as well as for the final classification fusion at the FC
- Code matrix designed to minimize the error probability of classification
- Two heuristic methods for code design (Wang *et al.*, 2005):
  - cyclic column replacement and
  - simulated annealing
- Exact expression characterizing the performance, depends on the application considered

Distributed *M*-ary Classification

- T.-Y. Wang, **Y. S. Han,** P. K. Varshney, and P.-N. Chen, "Distributed Fault-Tolerant Classification in Wireless Sensor Networks," ***IEEE Journal on Selected Areas in Communications (JSAC): special issue on Self-Organizing Distributed Collaborative Sensor Networks***, pp. 724-734, April, 2005.

Wireless Sensor Networks

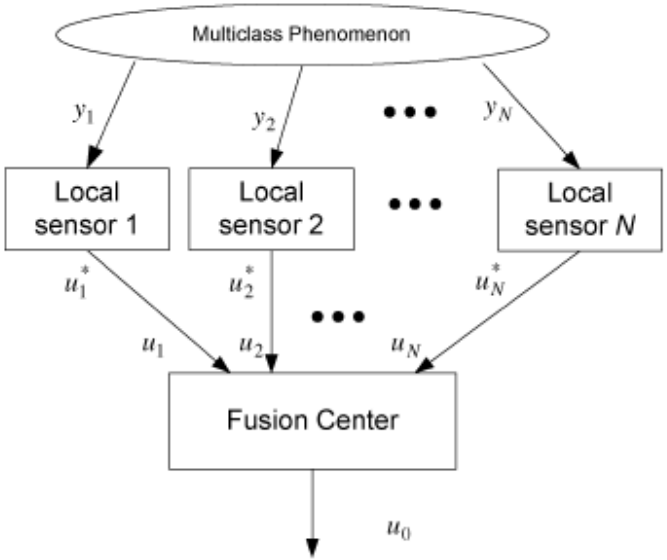
- Used in military and civilian application to monitor environment – detection, classification and/or estimation
- Bandwidth and Energy Constraints: Use Quantized data
- Performance depends on local sensor data
- Important to ensure reliable data
- Unreliable data due to faults, imperfect channels, and/or malicious sensors

Fault-tolerant Distributed Classification (Wang et al., 2005)

- Used for monitoring conditions like seismic activity or temperature
- Existence of faulty sensors deteriorate performance
- Straight-forward application of the DCFECC Approach
- Example: $C_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$
- True hypothesis is H_0
- Sensors 2 and 3 are faulty and stuck-at '1'
- Received vector: [1 1 1 0 1 0 1]
- Hamming distances are (2, 4, 5, 3) respectively
- Decide H_0 even with faulty sensors

Fault-tolerant Distributed Classification (contd..)

- Expression for error probability derived as a function of the code matrix $\boldsymbol{C}$
- Assign costs for misclassification of the proposed fusion rule
- Note that u_i^* and u_i may be different because of
  - permanent or transient faults
  - the malicious behavior of sensors
  - channel errors.

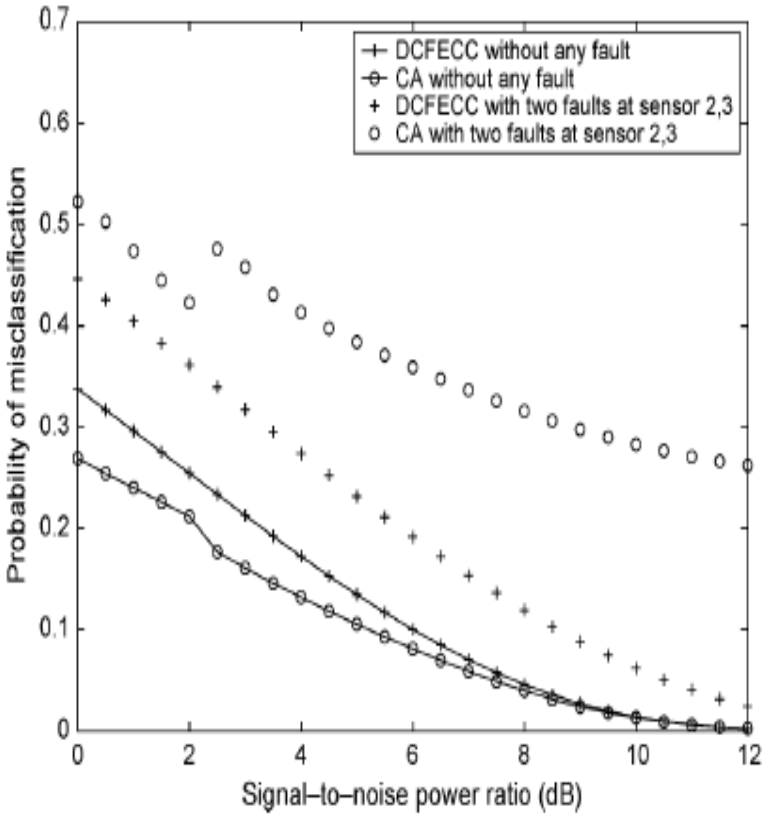


$$P_e = \sum_{i_1,\cdots,i_N,i_1^*,\cdots,i_N^*,l} \int_{y_1,\cdots,y_N} P_l P(u_1 = i_1 | u_1^* = i_1^*) \times \cdots \times P(u_N = i_N | u_N^* = i_N^*) P(u_1^* = i_1^* | y_1) \times \cdots \times P(u_N^* = i_N^* | y_N) P(y_1, \cdots, y_N | H_l) A^l_{i_1,\cdots,i_N},$$

Numerical Results

- $N = 7$ i.i.d. sensors performing a ($M = 4$)-ary classification
- Equally probable hypotheses Gaussian distributed hypotheses with different means
- Presence of stuck-at faults ('1') and transmission over ideal channels
- Simulated Annealing:

  $C_1 = [3, 8, 14, 12, 9, 12, 9]$
- Comparison with Conventional Approach using Chair-Varshney rule (Chair & Varshney, 1986)



Extensions

- Distributed Classification using Soft-Decision Decoding (DCSD) approach (Wang *et al.*, 2006):
  - non-ideal channels
  - use soft-decisions at the FC
  - reduce the errors due to channel uncertainties
- DCFECC using non-binary codes (Wang *et al.*, 2005)
- Sub-optimal code design schemes based on error bounds (Yao *et al.*, 2007)

Secure Target Localization

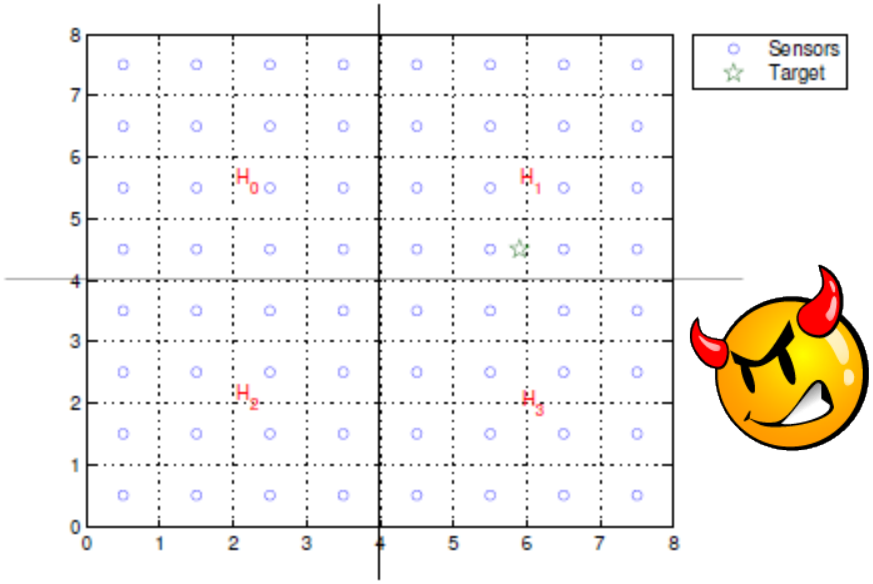
- A. Vempaty, Y. S. Han, and P. K. Varshney, "Target Localization in Wireless Sensor Networks using Error Correcting Codes," IEEE Trans. on Information Theory, pp. 697-712, January, 2014.

Wireless Sensor Networks- Revisit

- Task of target localization (Niu & Varshney, 2006)
- WSNs are prone to malicious attacks from within the network or outside
- Byzantine Attacks (Vempaty *et al.*, 2013):
  - Presence of Byzantine (compromised) nodes in the network
  - Send false information to the Fusion Center (FC)
  - Aim to deteriorate the performance of the inference process at the FC
- Goal:
  - Design energy efficient target localization scheme in WSNs using Error-Correcting codes
  - Tolerant to Byzantine data from the local sensors

Localization as Hierarchical Classification (Vempaty et al., 2014)

- Target emits power that follows an isotropic power attenuation model
- Local sensor i uses threshold quantizer (η_i) on its corrupted observation and decides D_i
- FC receives binary decision vector $\mathrm{u} = [\mathrm{u}_1, \cdots, u_N]$
- $\mathrm{u_i}$ need not be same as D_i due to the presence of Byzantine sensors



Localization as Hierarchical Classification (Contd..)

- Traditional approach: Maximum-Likelihood Estimator (MLE) based on the received data u
- Computationally very expensive: performs optimization over the entire region of interest (ROI)
- Computationally efficient method: model as hierarchical classification
- Splitting the ROI into *M* regions at every iteration and performing an *M*-ary classification to decide the ROI for the next iteration
- Classification at every iteration performed using the DCFECC approach
- Error-correction capability of the code matrix provides Byzantine fault-tolerance

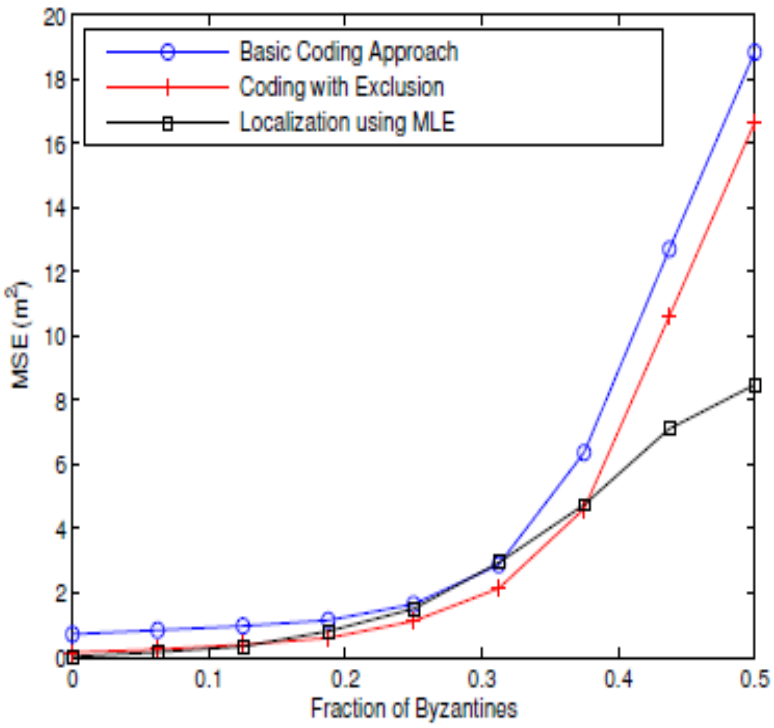
Code Design for the Scheme

- Scheme is hierarchical, the code matrix needs to be designed at every iteration
- Simple and efficient manner:
  - Size of $\boldsymbol{C}_{\mathrm{k}}$ at the $(k+1)^{\mathrm{th}}$ iteration is $M \times N/M^{k}$ where $0 < k < k^{stop}$; k^{stop} depends on the stopping criterion
  - Each row of $\boldsymbol{C}_{\mathrm{k}}$ represents a possible hypothesis described by a region (R_j^k) in the ROI
  - For j^{th} row, only sensors in R_j^k have '1' as their elements in the code matrix
- Example $\boldsymbol{C}_{\mathrm{k}}$: 16 sensors into 4 regions

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Numerical Results

- Exclusion-based scheme: store best 2 regions
- N=512 sensors deployed in 8 x 8 grid
- α is the fraction of Byzantines
- $N_{mc} = 1 \times 10^3$ Monte-Carlo runs to evaluate Mean Square Error (MSE)



Observations

- Performance of the exclusion-based coding scheme is better than the basic coding scheme
- Outperforms the traditional MLE based scheme when $\alpha \leq 0.375$.
- Proposed schemes are around 150 times faster than the conventional
- Computation time is very important when the target is moving and a coarse location estimate is needed in a timely manner

Extensions

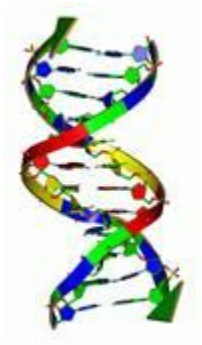
- Considered the effect of non-ideal channels (Vempaty *et al.*, 2014)
  - Suggested the use of soft-decision decoding similar to DCSD
  - Compensate for the loss due to the presence of fading channels between the local sensors and the FC
- Evaluated the performance of the proposed schemes in terms of the Byzantine fault tolerance capability and probability of detection of the target region (Vempaty *et al.*, 2014)
- Presented performance bounds which can be used for system design (Vempaty *et al.*, 2014)

Reliable Crowdsourcing

- A. Vempaty, L. R. Varshney, and P. K.Varshney, "Reliable Classification by Unreliable Crowds," in Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP2013), Vancouver, Canada, May 2013, pp. 5558–5562.

Humans vs Machines

- Current machines reduce human work
- But cannot completely replace them!
- Without proper “training”, machines cannot perform inference tasks reliably



Pattern Search

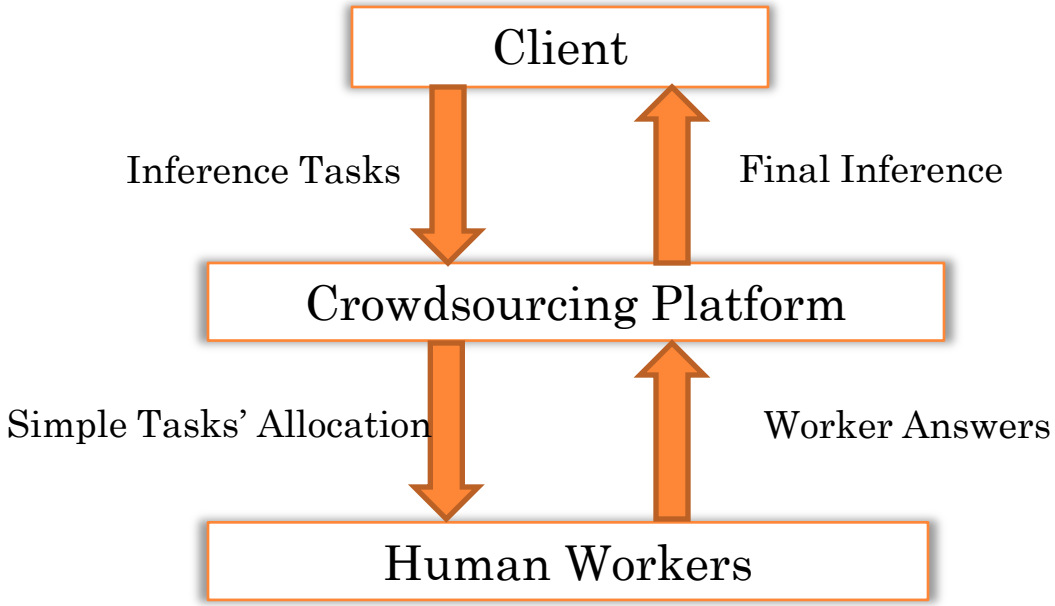
Data Interpretation

Crowdsourcing

- Crowd+Sourcing = Crowdsourcing
- New paradigm for human participation in distributed inference tasks

Crowdsourcing Challenges

- Key differences from team decision-making:
  - Number of participants involved in crowdsourcing are large
  - Members of the crowd are anonymous and may be unreliable or malicious
  - May not have sufficient domain expertise to perform full classification
- How to get reliable performance? and how to design the questions?



Problem Formulation

- **Focus**: Classification task consisting of M classes: $H_0, H_1, \cdots, H_{M-1}$
- **Goal**: Design questions for the N crowd workers to ensure reliable classification
- Easy for crowds to answer binary questions (Branson *et al.*, 2010)

Dog breed?

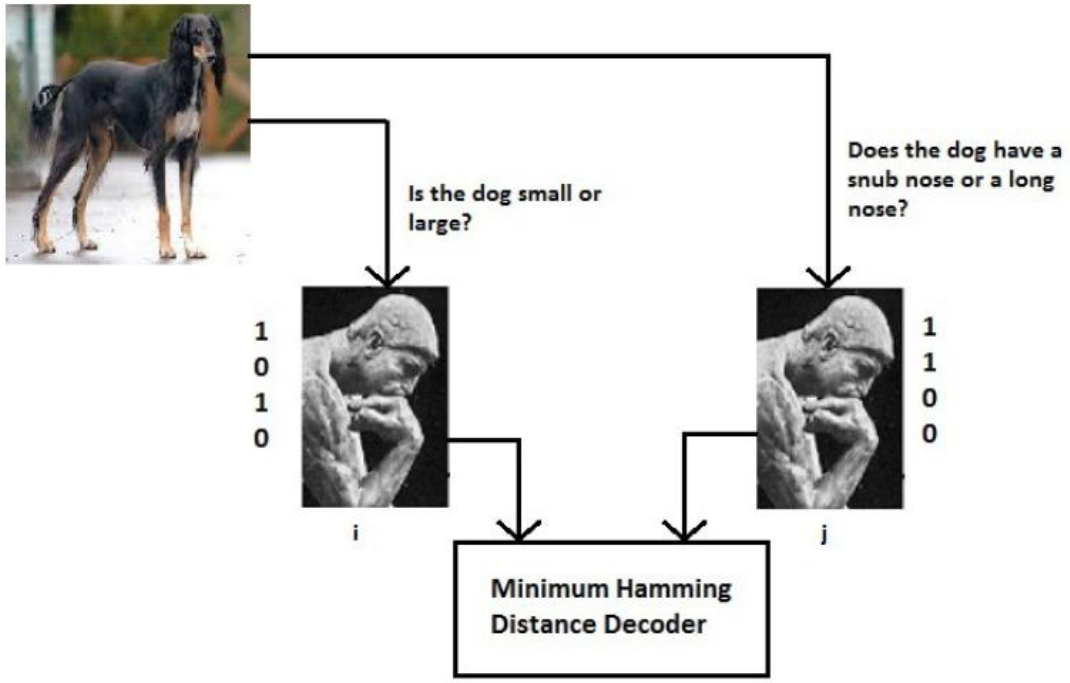
Snub or long nose?

Coding for Crowdsourcing (Vempaty et al., 2013)

- Design of questions ⇔ Design of $M \times N$ binary code matrix $A = \{c_{li}\}$
- Rows of C correspond to the different classes
- Column c_i corresponds to the question to the i^{th} worker
- Code matrix designed to minimize misclassification probability
- Code design is based on performance evaluation

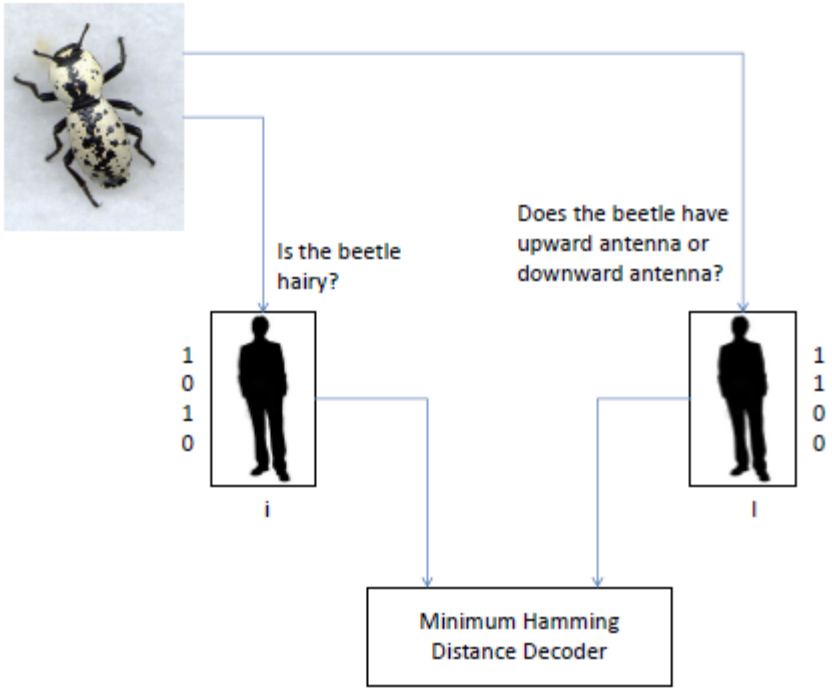
Example 1

- Task: Classification of dog image into one of four breeds: Pekingese, Mastiff, Maltese, or Saluki
- Let the columns corresponding to the i^{th} and j^{th} workers be $c_i = [1010]'$ and $c_j = [1100]'$ respectively



Example 2

- Task: Classification of beetle image into one of four breeds: Drug Store Beetle, Apricot Beetle, Variegated Mud-loving Beetle, or Ironclad Beetle
- Let the columns corresponding to the i^{th} and j^{th} workers be $c_i = [1010]'$ and $c_j = [1100]'$ respectively



Worker Model

- Worker j decides the true class (local decision y_j) with probability p_j and the wrong local classification with uniform probability:

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

- Anonymous crowds, so specific reliability cannot be identified
- Assume that each worker j in the crowd has an associated reliability p_j
- Reliability modeled as a random variable to capture the workers randomness
- Two i.i.d. crowd models: spammer-hammer and Beta model
  - Spammer-Hammer: spammers have $p_j = 1/M$ and hammers have $p_j \to 1$
    - Quality of the crowd, Q, is governed by the fraction of hammers
  - Beta model: p_j follows Beta distribution

Classification Performance

- Performance in terms of average error probability for classification
- Compare the coding approach to the traditional majority approach
- Majority voting:
  - N workers are split into $\log_2 M$ groups with each group sending information regarding a single bit
  - Majority rule to decide each of the $\log_2 M$ bits separately
- Notations:
  - Reliabilities: $\boldsymbol{p} = [p_1, p_2, \ldots, p_N]$ i.i.d. random variables with mean μ
  - Crowdsourcing system: (N, M, μ)
- Performance improves with increasing value of μ

System Characterization

- 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 $\mu > \mu'$.

- Performance criterion is average error probability; weak criterion of crowd-ordering in the mean sense
- Better crowds yield better performance in terms of average error probability

System Characterization

- For N = 10 workers and M = 4 classes, good code matrix is found by simulated annealing (Wang *et al*. 2005)

$$C_1 = [5, 12, 3, 10, 12, 9, 9, 10, 9, 12]$$

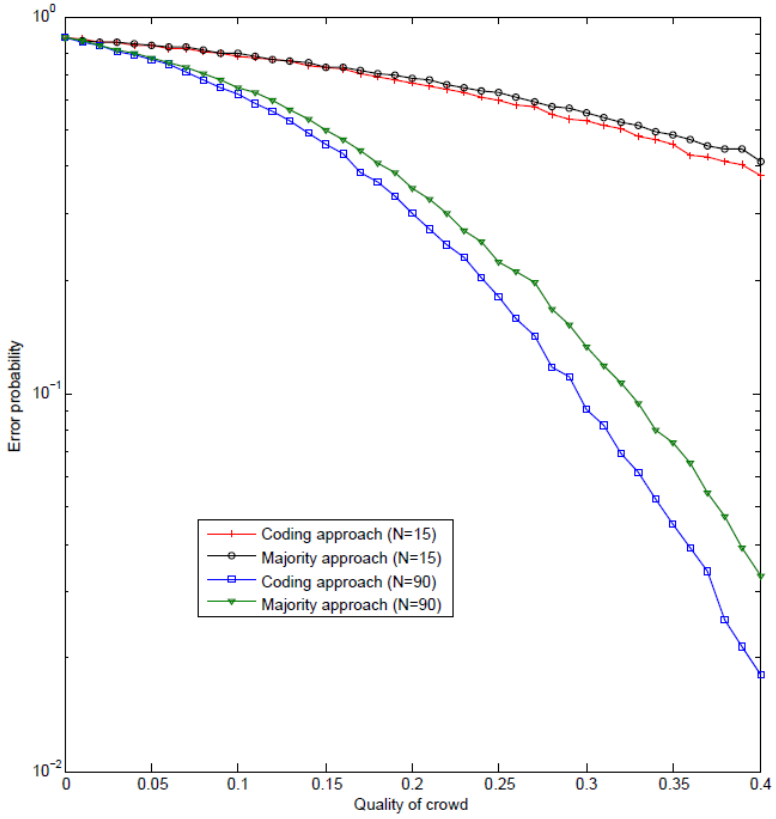
- For larger system, N = 15 workers and M = 8 classes, good code matrix is found by cyclic column replacement (Wang *et al*. 2005)

$$C_2 = [150, 150, 90, 240, 240, 153, 102, 204, 204, 204, 170, 170, 170, 170, 170]$$

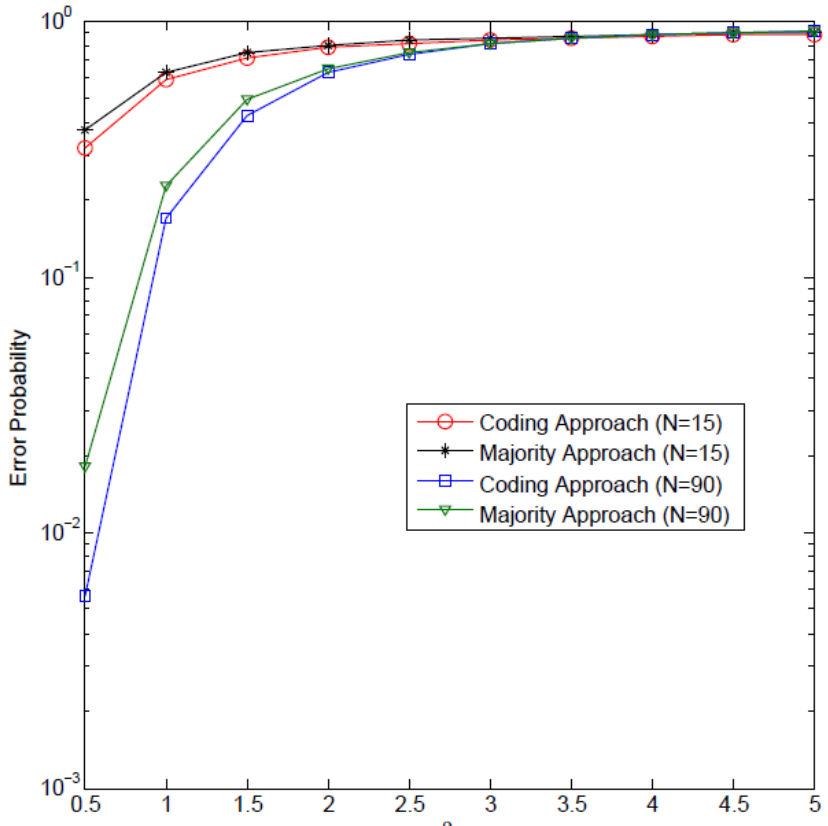
- For a system consisting of N = 90 workers and M = 8 classes, sub-optimal code matrix by concatenating the columns of C_2.

Coding is better than Majority Voting

- Gap in performance generally increases for larger system size
- Good codes perform better than majority vote as they diversify the binary questions



Spammer-hammer model, (M = 8; varying N)



Beta(α = 0.5, β) model, (M = 8; varying N)

Experimental Results

- Tested the proposed coding approach on six publicly available Amazon Mechanical Turk data sets of affective text task[1]
- 100 tasks with $N = 10$ workers taking part in each of the tasks
- Quantize dataset values by dividing the range into $M = 8$ equal intervals
- Proposed approach outperforms the majority approach in 4 out of the 6 cases considered

Fraction of errors using coding and majority approaches

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

[1]http://ai.standford.edu/~rion/annotations/

Implications

- Coding approach can more efficiently use human cognitive energy over traditional majority-vote methods
- Very useful for applications where number of classes are large:
  - Fine-grained image classification for building encyclopedias like Visipedia where one might need to classify among more than 161 breeds of dogs or 10000 species of birds
- Designing easy-to-answer binary questions using the proposed scheme greatly simplifies the workers' tasks

Extensions

- Extend to other crowdsourcing models (Vempaty *et al.*, under review):
  - Effect of social aspects of workers such as coordination or competition which result in correlated reliabilities
  - Common sources of information, where the worker observations are dependent
- Can better cognitive and attentional models of human crowd workers provide better insight and design principles?

Conclusion

- Coding theory based techniques can be used to ensure reliable signal processing
- DCFECC can be used in various signal processing applications to handle erroneous data from agents
- Many other applications fit this generalized framework where reliable processing could be ensured by DCFECC
- For example, system consisting of agents who would have some elements of human computation models and some elements of WSN models

QUESTIONS?