AMSC 663 Proposal: Task Assignment in a Human-Autonomous Image Labeling System

Addison Bohannon

October 14, 2015

Advisors:

Vernon J. Lawhern
Human Research and Engineering Directorate
US Army Research Laboratory
Aberdeen Proving Ground, MD 21005 USA
*v*ernon.j.lawhern.civ@mail.mil

Brian M. Sadler
Computational and Information Sciences Directorate
US Army Research Laboratory
Adelphi Laboratory Center, MD 20783 USA
*b*rian.m.sadler6.civ@mail.mil

Abstract

We want to design a human-autonomous system which can efficiently and accurately classify a database of images as interesting or not interesting. The system will iteratively distribute images for binary classification amongst multiple hetergeneous agents according to the Generalized Assignment Problem, and use the Spectral Meta-Learner, to combine multiple binary classification results in a principled manner. From the results, we hope to draw conclusions about the circumstances under which sensitive tasks can and should be delegated to autonomous technology in an optimal manner.

1 Introduction

Suppose that we wanted to sort a large database of images according to a simple binary classification such as interesting or not interesting. The defense and intelligence communities have this exact requirement. The sorting of satellite imagery requires trained professionals to scan images for indicators of valuable intelligence, triaging images for later thorough analysis. This is a costly and tedious task for humans. Certainly, there are means to reduce the workload for those intelligent analysts in the age of computer vision. However, when considering tasks with implications as serious as our national security, perhaps we cannot exclude humans from the decision loop.

Sajda, et al. explore this very problem in [11]. They used computer vision algorithms to complement the efforts of a human performing image triage, sorting images as "interesting" or not. The group sought to synchronize the efforts of a single human performing Rapid Serial Visual Presentation (RSVP) with a single computer vision system. [1]

Exploiting the singular ability of humans to quickly understand the "gist" of an image, the human agent quickly scanned the images. Images which elicited interesting responses through the RSVP paradigm were routed to the computer vision algorithm, which excels at object recognition. In essence, the system could intelligently sort the images according to the human analysis, and determine the source of the humans interest through the far quicker and more accurate object recognition skills of computer vision. Sajda, et al. additionally implemented this system in reverse, pruning the image database first through computer vision algorithms searching the images for predetermined interesting objects and routing the reduced database to the human for confirmation. Either method, autonomous technology as a mechanism for inference or as a filter, proved a viable complement to the single image analyst, and the results imply a more general approach to leveraging autonomous technology to support human efforts.

An obvious concern might be a lack of sensitivity of a computer vision algorithm. What if a valuable image were deemed not valuable and thus never seen by a human analyst? Here, Ipierotis, et al. present a sensible recommendation: seek additional independent classifications from other agents when the classification outcomes of tasks have a non-negligible probability of error [5]. If one computer vision algorithm deems an image as interesting, confirm with another computer vision algorithm – or better yet, confirm with as many distinct agents as feasible. This implies that in order to reduce the workload for the human imagery analyst, multiple autonomous agents must make independent classifications, requiring the human agents input only after the joint classification of autonomous agents does not provide a confident meta-classification.

Such a system is depicted in Figure 1; however, it begs two principal questions: how should images be distributed amongst the agents in such a system, and how should multiple labels for a single image be combined to make a decision about that image?

1.1 Problem

Let us simplify and formalize this system in order to formulate a clear problem statement. Figure 2 conceptualizes a system with n heterogeneous tasks and m heterogeneous agents. Let T be the set of tasks. Then an agent performs the following binary decision: $Y : T \to \{-1, 1\}$. Denote $I = \{1, \ldots, n\}$ as the index of tasks and $J = \{1, \ldots, m\}$ as the index of agents. The assignment of a task $i \in I$ to an agent $j \in J$ will have an associated value, $v_{ij} = f(s_i, r_j) \geq 0$. Each task has an associated score parameter, $0 \leq s_i \leq 1$, which reflects the current confidence in the meta-label

[1] RSVP is a Brain-Computer Interface paradigm in which subjects wear an electroencephalogram (EEG) and passively view images at high rates of speed (2-5 Hz). Using the oddball paradigm, novel stimuli, or interesting images, generate a neural signature in the EEG recordings which can be recognized through machine learning methods.

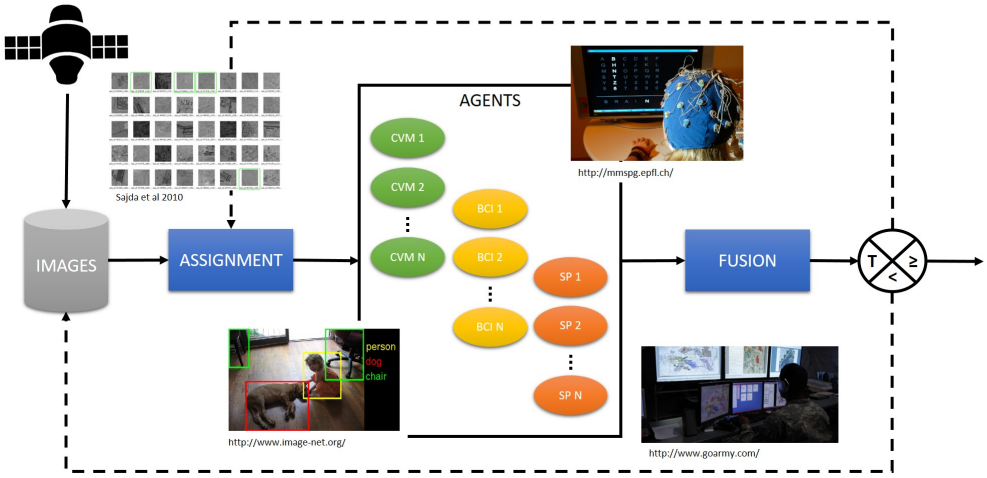

Figure 1: The image labeling system. An imaging sensor populates the database of images. An assignment node distributes images to agents in parallel. The agents perform binary classification – interesting or not interesting, and the results are consolidated at a fusion node. At this point, the confidence in the image classification label is used to threshold images for completion or routing back to the image database for re-assignment. Here, agents can be computer vision algorithms, RSVP subjects, or self-paced image analysts.

of that task. Each agent has a reliability, r_j, which reflects the agents accuracy, a cost, c_{ij}, which could reflect the time or resources required for task completion and may depend on the task, and a budget, b_j, which limits the workload for an agent on each iteration.

Accordingly, we want to determine:

1. Assignment Problem – How do we optimally assign images in each iteration in order to maximize value?
2. Joint Classification Problem – How do we infer the label of a task given a set of labels from multiple agents?

1.1.1 Task Assignment

The task assignment problem can be mapped onto the generalized assignment problem (GAP) as follows, where we are looking for the optimal assignment, captured by, $\mathbf{x}_{ij}$, the assignment of task i to agent j [8, 7]:

$$G = \min_{\mathbf{x}} \sum_{i \in I} \sum_{j \in J} -v_{ij} \mathbf{x}_{ij} \tag{1}$$

1. $\sum_{i \in I} c_{ij} \mathbf{x}_{ij} \leq b_j, \ j \in J$
2. $\sum_{j \in J} \mathbf{x}_{ij} = 1, \ i \in I$
3. $\mathbf{x}_{ij} \in \{0, 1\}$
4. $v_{ij} = g(r_j, s_j) \geq 0$

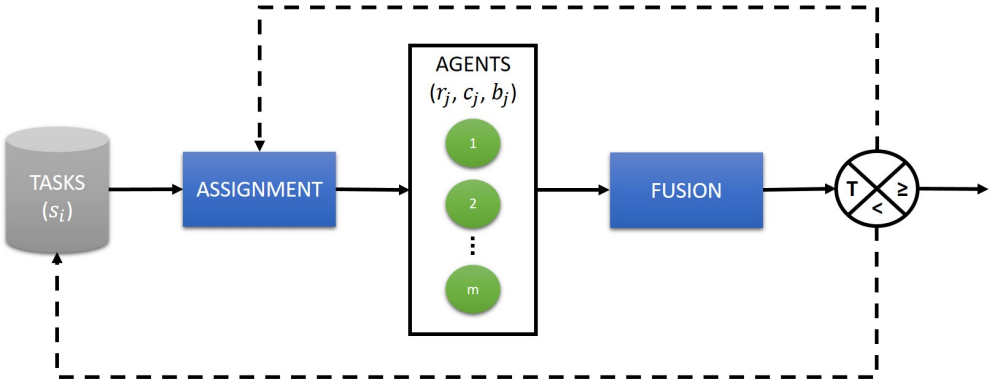

Figure 2: This iterative task assignment system is a simplification of the image labeling system. Here, heterogeneous tasks, with an extrinsic confidence score, s_i, are assigned in parallel from a centralized node to heterogeneous agents, with the following intrinsic parameters: reliability, r_j, and budget, b_j. Additionally, each assignment has a cost, c_{ij}. The classification results are consolidated into a single joint classification at the fusion node, and the images are classified if the confidence in the meta-label exceeds a pre-set threshold, otherwise images are routed back to the database for re-assignment.

GAP is a NP-hard binary integer problem with two classes of solutions – exact and heuristic. Exact methods seek to exhaustively search the solution space for a global minimum; however, the most successful exact methods do not evaluate the target function at all feasible solutions, but rather use a heuristic to limit the feasible solutions for search. As the GAP is combinatorial, computational complexity of even the best exact methods can be prohibitive. For that reason, researchers often apply heuristic methods [7, 8].

A proven exact method for solving the GAP is the Branch and Bound algorithm (B&B). B&B is a divide and conquer optimization approach with a bounding function, search strategy, and branching strategy. The algorithm searches branches, or sub-problems of the overall problem and calculates bounds on the sub-problems according to heuristic methods. If the bounds are sub-optimal, or above the current optimal solution, then the algorithm can prune these sub-problems and search elsewhere. Otherwise, this sub-problem branches into further sub-problems for search. The algorithm iterates until the problem space is exhausted, and a global optimum is achieved. See Algorithm 1 for B&B pseudo-code.

1.1.2 Joint Classification

Given m agents which independently produce binary classifications on n tasks, the joint classification problem asks how do we obtain an improved estimate of the label of each task without direct knowledge of the agent performance probabilities and without knowledge of the true labels of any of the tasks? We seek to minimize the probability of error by finding a mapping, f, from a set of the independent classification labels from m agents to a meta-label such that the probability over all tasks of the meta-label being incorrect is minimized:

$$\arg\min_{f} \sum_{i \in I} \mathbb{P}(f(\hat{y}_{i1}, \ldots, \hat{y}_{im}) \neq y_i) \tag{2}$$

1. $f : \{\hat{y}_{ij}\}_{j \in J} \rightarrow \bar{y}_i$

2. $\hat{y}_{ij},\ \bar{y}_i,\ y_i \in \{-1, 1\}$

The classificaiton labels of all agents for all tasks can be captured in a matrix, $\mathbf{A} \in \{-1, 1\}^{m \times n}$, in which the rows correspond to agents and the columns correspond to tasks, see Figure 3.

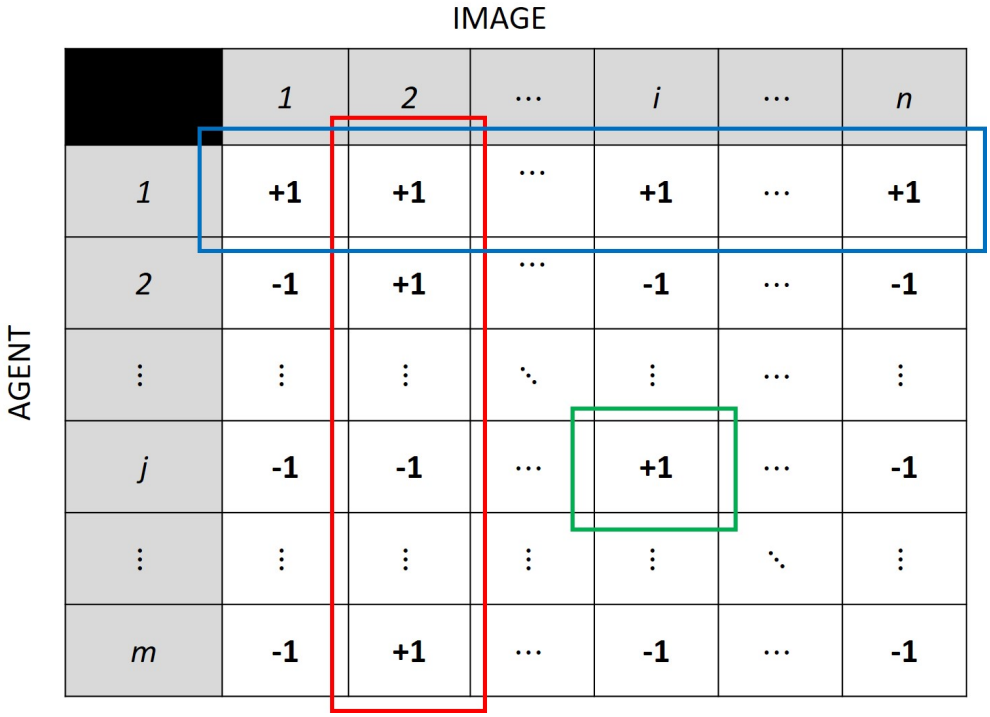

Figure 3: The classification outcome matrix, $\mathbf{A}$, captures the meta-data across all agents and tasks. The green box indicates entry A_{ij} which coincides with $\hat{y}_{ij}$ the outcome of agent j classifying task i. The red rectanble comprises the label set, $\{\hat{y}_{ij}\}_{j \in J}$, for a task, i. The blue rectangle highlights a row vector of the label set for an agent j, referred to as $\mathbf{a}_j \in \{-1, 1\}^n$ in (9).

Approaches to the problem of joint classification fall into one of three categories: supervised, unsupervised, and semi-supervised. Supervised methods rely on ground-truth knowledge of a portion of the task labels in order to learn an optimal operation, f, and semi-supervised methods have access to a source of ground-truth knowledge for task labels so that it can be attained when appropriate. For unsupervised methods such as our problem, labels are typically inferred according to majority vote or averages [5, 9].

1.2 Relevant Work

Much of the related work to this problem deals with optimal crowdsourcing, in which binary classification tasks are distributed in parallel to numerous agents at low-cost. In practice, on services such as Amazon Mechanical Turk, the responses of agents can be completely random, and it is desirable to learn a reliable binary classification for numerous tasks from the labels of noisy labelers [5]. Like our problem, it requires the assignment of binary classification tasks to noisy labelers and the need for joint classification of repeated labels from different agents for the same task.

In a very simplistic model, Karger, et al. explore the optimal crowdsourcing problem within a framework of homogeneous agents and homogeneous tasks without supervised knowledge of any task labels. In this special case, no assignment strategy can do better than random assignment, and they instead develop a belief propagation method for inferring agent performance to perform the joint classification [6].

Ho, et al. developed an adaptive task assignment method for crowdsourcing problems, framing the problem in very similar terms to our problem formulation [4]. They approached the assignment of heterogeneous binary tasks to heterogeneous agents within the GAP framework. Unlike in our approach, they did not constrain the joint classification to be unsupervised. Instead, they proposed an exploration-exploitation method in which additional agents were recruited to infer the difficulty of tasks, and a subset of known tasks were assigned to infer the agent accuracy, similar to a training session.

The question of joint classification without supervised knowledge of agent performance probabilities or true labels for any images highlights the particular benefits of the Spectral Meta-Learner (SML) introduced by Parisi, et al. [9]. SML provides a fully unsupervised method for not only joint classification but also estimating the accuracy of agents; however, in [9], the method is only applied in post-hoc analysis. In our system, we will apply it adaptively, using the estimation of agent accuracy to make further task assignments.

Solving GAP with B&B is well-established in the literature. We will follow the Lagrangian Relaxation (LR) implementation of Fisher, et al. in [2, 3]. Considering the scale of our problem, dualizing the semi-assignment constraint and using the Lagrange multiplier adjustment method heuristic as in [3] will likely provide better computational efficiency than subgradient methods either of the dual problems.

2 Approach

2.1 Branch and Bound Algorithm

We will implement B&B at the assignment module to solve the generalized assignment problem. For the bounding function, we will use LR of the semi-assignment constraints with either subgradient methods [2] or multiplier adjustment methods [3]. For the search strategy, a best first search will be used. B&B pseudo-algorithm is shown in Algorithm 1 [1].

Algorithm 1: Branch and Bound

Data: $S \subset \{0,1\}^{m \times n}$, $g: \ S \to \mathbb{R}$
Result: $x_{opt} \in \{0,1\}^{m \times n}$
$I := \infty$; $LB(p_0) := g(p_0)$; $B := \{(p_0, LB(p_0))\}$;
while $B \neq \emptyset$ **do**
  Select $p \in B$; $B := B \backslash \{p\}$; branch on p for $1, \ldots, k$;
  for $i = 1, \ldots, k$ **do**
    $LB(p_i) := g(p_i)$;
    if $LB(p_i) < I$ **then**
      if $LB(p_i) = f(X)$ **then**
        $I = f(X)$; $x_{opt} = X$; go to end;
      else
        $B := B \cup \{(p_i, LB(p_i))\}$;
      end
    end
  end
end

2.1.1 Lagrangian Relaxation

The crux of an efficient solution to GAP using B&B is having a sufficiently tight bounding function. For this purpose, we will use LR, which has been demonstrated to provide a sufficiently tight bound for B&B for binary integer problems of our problem size [2, 3, 10].

Considering the set of constraints on GAP, we face a choice of which constraint subset to relax. One option is to relax the capacity constraints (constraint 1 in (1)) as described in [2]:

$$L^c(\lambda) = \text{minimize} \left(\sum_{i \in I} \sum_{j \in J} -v_{ij} \mathbf{x}_{ij} + \sum_{j \in J} \lambda_j \sum_{i \in I} (c_{ij} \mathbf{x}_{ij} - b_j) \right) \tag{3}$$

1. $\sum_{j=1}^{m} \mathbf{x}_{ij} = 1, \ i \in I$
2. $\mathbf{x}_{ij} \in \{0,1\}$
3. $\lambda_j \geq 0, \ j \in J$

The problem is now easier owing to the reduced number of hard constraints, but more than this improvement, we want to leverage the intrinsic structure of the problem and relax constraints which will yield a substantially easier problem. As shown in [3], another option is to relax the semi-assignment constraints (constraint 2 in (1)):

$$L^a(\lambda) = \text{minimize} \left(\sum_{i \in I} \sum_{j \in J} -v_{ij} \mathbf{x}_{ij} + \sum_{i \in I} \lambda_i \sum_{j \in J} 1 - \mathbf{x}_{ij} \right) \tag{4}$$

1. $\sum_{i \in I} c_{ij} \mathbf{x}_{ij} \leq b_j, \ j \in J$
2. $\mathbf{x}_{ij} \in \{0,1\}$

Notice that the problem now simplifies to m knapsack problems, a well-studied combinatorial optimization problem:

$$L_j^a(\lambda) = \text{minimize} \left(\sum_{i \in I} (-v_{ij} + \lambda_i) \, \mathbf{x}_{ij} \right), \; j \in J. \tag{5}$$

In [3], Fisher et al show decreased time complexity with (4) over (3). Additionally, Morales et al prove the following inequality which shows that the bound from (4) is tighter than the bound from (3) [8]:

$$\max L^c(\lambda) \leq \max L^a(\lambda) \leq G. \tag{6}$$

where G is the solution to the primal problem.

2.1.2 Subgradient Method

Let $Z = \max_{\lambda} L^a(\lambda)$. Z is a relatively tractable optimization problem with the exception of not being everywhere differentiable, which precludes the exclusive use of gradient based optimization methods [2]. This problem can be overcome with the use of subgradient methods. A subgradient of $L^a(\lambda)$ at λ_0 is a vector, ν, such that

$$L^a(\lambda) \leq L^a(\lambda_0) + \nu(\lambda - \lambda_0), \; \forall \; \lambda. \tag{7}$$

Fortunately, Z, is sub-differentiable everywhere, and as with gradient based methods, the optimal solution to Z occurs when 0 is a subgradient of $L^a(\lambda)$ [2], and we can take iterative steps along subgradients to find a 0 subgradient according to the following iteration rule from [2]:

$$\lambda^{k+1} = \lambda^k + \frac{\alpha(UB(Z) - L^a(\lambda^k))}{\|\vec{1} - x^k\|^2}(\vec{1} - x^k), \tag{8}$$

where $\vec{1} = (1, \ldots, 1)^T$, $(\vec{1} - x^k)$ is the subgradient, $0 < \alpha \leq 2$, $UB(\cdot)$ is a heuristic upper bound, and C, x, and b are vectorized components of (4). This iterative procedure guarantees convergence to $\max_{\lambda} L^a(\lambda)$ [2].

2.1.3 Multiplier Adjustment Method

Another approach to solving Z is the multiplier adjustment method, a heuristic technique. In the multiplier adjustment method, we are searching the solution space similar to a steepest descent method; however, we only search the solution in constrained directions, changing a small, fixed subset of the Lagrangian multipliers, λ_i, one at a time. This greatly increases the speed and decreases the computational cost of each gradient step, as we can simply find the maximum directional derivative at each iteration [2]. The multiplier adjustment method is not guaranteed to find a global optimum of the dual problem, Z; however, practice, it converges considerably quicker than subgradient methods, without sacrificing accuracy [3].

We will follow the algorithm proposed in [3] for implementation of the multiplier adjustment method (see Algorithm 2 for pseudo-code).

Algorithm 2: Multiplier Adjustment Method

Data: $v_i(\lambda) = (\lambda_i - v_{i1}, \ldots, \lambda_i - v_{im})$, $x_j = (x_{1j}, \ldots, x_{nj})$, $x = (x_1, \ldots, x_m)$
Result: Z
$u_j = \min_2 v_i(0)$; $x_{ij} = 0$; $I_j^+ = \{i \in I | \lambda_i - v_{ij} > 0\}$;
$\mathbf{x} = \arg\min_{\mathbf{x}} \sum_{i \in I_j^+} (\lambda_i - v_{ij})\, \mathbf{x}_{ij}$, $\sum_{i \in I_j^+} c_{ij} x_{ij} \leq b_j$, $x_{ij} \in \{0,1\}$, $i \in I_j^+$;
while do
  $\bar{I} = \{i \in I | \sum_{j \in J} x_{ij} = 0\}$; $I_j^0 = \{i \in I | \lambda_i - v_{ij} = 0\}$; $J_i^0 = \{j \in J | i \in I_j^0\}$;
  $\bar{b}_j = b_j - \sum_{i \in I} c_{ij} x_{ij}$;
  $\mathbf{x} = \arg\min_{\mathbf{x}} \sum_{j \in J_i^0} \sum_{i \in I_j^0} -v_{ij} \mathbf{x}_{ij}$, $\sum_{j \in J_i^0} x_{ij} = 1$, $i \in \bar{I}$, $\sum_{i \in I_j^0} c_{ij} x_{ij} \leq \bar{b}_j$, $j \in J$, $x_{ij} \in \{0,1\}$,
  $j \in J_i^0$, $i \in I_j^0$;
  if $\sum_{j \in J} x_{ij} = 1$, $i \in I$ **then**
    return ;
  end
  for $i = \{i \in I | \sum_{j \in J} x_{ij} = 0\}$ **do**
    for $j \in J$ **do**
      $\delta_{ij} = Z_j(u) + v_{ij} - \lambda_i + \min \sum_{l \in I | l \neq i} (\lambda_l - v_{lj}) x_{lj}$, $\sum_{l \in I | l \neq i} c_{lj} x_{lj} \leq b_j - c_{ij}$, $x_{lj} = \{0,1\}$,
      $l \in I / \{i\}$;
      $I^0 = \{i \in I | \sum_{j \in J} x_{ij} = 0, \min_2(\delta_{i1}, \ldots, \delta_{im}) > 0\}$;
    end
  end
  if $J^0 = \emptyset$ **then**
    return;
  else
    Choose $i* \in I^0$; $j* = \arg\min(\delta_{i*1}, \ldots, \delta_{i*m})$; $x_{i*j*} = 1$;
    $\min \sum_{i \in I | i \neq i*} (\lambda_i - v_{ij}) x_{il}$, $\sum_{i \in I | i \neq i*} c_{ij} x_{ij} \leq b_i - c_{i*j}$, $x_{ij} = \{0,1\}$, $i \in I / \{i*\})$;
    $I^0 - I^0 / \{i*\}$;
  end
end

2.2 Spectral Meta-Learner

Under reasonable assumptions – independent and identically distributed task classification and conditionally independent classification from agents – the work of Parisi, et al., allows us to develop an unsupervised weighting of agents according to the balanced accuracy, $\pi_j = \frac{1}{2}(\psi_j + \eta_j)$, of each of the m agents as in [9]. Here, ψ is sensitivity, $\psi_j = \mathbb{P}(\hat{y}_{ij} = 1 | y_i = 1)$, and η_j is specificity, $\eta_j = \mathbb{P}(\hat{y}_{ij} = -1 | y_i = -1)$.

Consider the sample covariance matrix of predicted labels over n tasks.

$$\hat{\mathbf{Q}} = \frac{1}{m-1}\sum_{j\in J}(\mathbf{a}_j - \bar{\mathbf{a}})^T(\mathbf{a}_j - \bar{\mathbf{a}}), \tag{9}$$

where a_j is the row of $\mathbf{A}$ (See Figure 3). As shown in [9], in the limit, as $n \to \infty$, the sample covariance matrix, $\hat{\mathbf{Q}} \in \mathbb{R}^{m\times m}$ will approach:

$$Q_{ij} = \lim_{n\to\infty} \hat{Q}_{ij} = \begin{cases} 1-\mu_j^2 & i = j \\ (1-b^2)(2\pi_i - 1)(2\pi_j - 1) & \text{o.w.} \end{cases}. \tag{10}$$

This implies that the covariance matrix will be nearly rank one, $\mathbf{Q} \approx \lambda\mathbf{v}\mathbf{v}^T$, and further that $v_j \propto (2\pi_j - 1)$. The entries of the principal eigenvector will be proportional to the balanced accuracy of the corresponding agent!

Although, we do not have the knowledge at hand to find $\mathbf{Q}$ analytically, we can use the sample covariance matrix as an estimate and simplify the problem of finding the principal eigenvector of $\mathbf{Q}$ to finding the principal eigenvector of a well-estimated sample covariance matrix, $\hat{\mathbf{Q}}$.

Further developed in [9], in a maximum likelihood formulation, the most likely label for a task, $\bar{y}_i$, is

$$\bar{y}_i = \text{sign}\left(\sum_{j=1}^{m} \hat{y}_{ij}\left(\log\left(\frac{\psi_j\eta_j}{(1-\psi_j)(1-\eta_j)}\right) + \log\left(\frac{\psi_j(1-\psi_j)}{\eta_j(1-\eta_j)}\right)\right)\right), \tag{11}$$

and upon a Taylor series expansion of $\bar{y}_i$ about $(\psi_j, \eta_j) = (1/2, 1/2)$, we have

$$\bar{y}_i \approx \text{sign}\left(\sum_{j=1}^{m} \hat{y}_{ij}\cdot(2\pi_j - 1)\right) \approx \text{sign}\left(\sum_{j=1}^{m} \hat{y}_{ij}\cdot v_j\right). \tag{12}$$

This means that the best mapping, f, to jointly classify a task in a maximum likelihood formulation is a weighted linear combination of agent classification labels where the weights correspond to the accuracy of each agent. Moreover, the accuracy of each agent can be estimated by calculating the principal eigenvector of the sample covariance matrix.

3 Implementation

Building a functioning on-line system will require minimizing the time and space complexity of the assignment module and implementing a robust message-passing system between assignment, fusion, and agent objects. The time complexity of B&B can be prohibitive for an assignment problem with too many agents and images. This requires exploring the optimization technique used to solve the LR problem, (4). Subgradient methods provide a more principled optimization technique but incur a greater cost in complexity. Multiplier reduction methods provide an approximation of the optimal solution but offer savings in time.

3.1 Resources

All software will be developed in MATLAB R2015b. Validation of the task assignment module will be run on a Windows-based laptop computer with an Intel Core i7 2.6 GHz processor and 8GB RAM. The testing will be run on a Unix-based desktop computer with 16 Intel Core i5 2.0 GHz processors and 100GB RAM.

3.2 Validation

Since the fusion module has been developed and validated during previous work, only the assignment module will require validation. Our developed code will be compared against the internal MATLAB mixed integer programming function (*intlinprog*()) in the Optimization Toolbox. Using known image confidence and agent reliability, we can verify that the application will yield equivalent assignment solutions by comparing the the optimal values of G.

3.3 Testing

In order to test the system, we will use the off-line classification results of computer vision algorithms and RSVP subjects on a common image database. These off-line results will be used to simulate on-line performance of the system. Since this system should attempt to match the accuracy of assigning all images to all agents while decreasing the overall number of task assignments, we will compare the system run time and accuracy of the system versus assignment of all images to all agents.

3.4 Databases

For testing, we will use the classification on a common image database, the Office Object Database, maintained by the Translational Neuroscience Branch of the Army Research Laboratory. The Office Object Database comprises a collection of images of office objects in natural and synthetic environments which pose object detection and recognition issues for both humans and computer vision. For instance, objects may be partly occluded and thus difficult for computer vision to recognize, but also objects may be un-centered in the image which makes detection during RSVP very difficult. Experimental data has been collected on RSVP subjects prompted to look for a particular office object such as a desk. Computer vision results will be collected similarly, requiring a binary response from computer vision algorithms based on the presence of a prompted office object.

3.5 Deliverables

- Software
  - Image Labeling System (fusion module, assignment module, agent classes)
  - Execution script
- Data
  - Office Object Database
  - Office Object RSVP Database
- Analysis
  - Performance analysis of test results
  - Implications for human-autonomous systems

3.6 Schedule

- Develop Assignment Module (15 OCT - 4 DEC)
  - Implement branch and bound algorithm (6 NOV)*
  - Validate branch and bound algorithm (25 NOV)*

  - Implement greedy search algorithm
  - Mid-year Review (4 DEC)*
- Build Image Labeling System (25 JAN - 26 FEB)
  - Build agent classes
  - Develop message-passing framework
  - Integrate all components into a system (26 FEB)*
- Test Image Labeling System (26 FEB - 15 APR)
  - Testing (1 APR)*
  - Performance analysis of test results
- Conclusion (15 APR - 1 MAY)
  - Final Presentation and Results (6 MAY)*

(* Denotes a milestone)

References

[1] Jens Clausen. Branch and bound algorithms-principles and examples. 1999.

[2] Marshall L. Fisher. The Lagrangian Relaxation Method for Solving Integer Programming Problems. *Management Science*, 50(12_supplement):1861–1871, December 2004.

[3] Marshall L. Fisher, R. Jaikumar, and Luk N. Van Wassenhove. A Multiplier Adjustment Method for the Generalized Assignment Problem. *Management Science*, 32(9):1095–1103, September 1986.

[4] Chien-ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. *Adaptive Task Assignment for Crowdsourced Classification.*

[5] Panagiotis G. Ipeirotis, Foster Provost, Victor S. Sheng, and Jing Wang. Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery*, 28(2):402–441, March 2013.

[6] David R. Karger, Sewoong Oh, and Devavrat Shah. Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems. *Operations Research*, 62(1):1–24, February 2014.

[7] O. Erhun Kundakcioglu and Saed Alizamir. Generalized assignment problem Generalized Assignment Problem. In Christodoulos A. Floudas and Panos M. Pardalos, editors, *Encyclopedia of Optimization*, pages 1153–1162. Springer US, 2008.

[8] Dolores Romero Morales and H. Edwin Romeijn. The Generalized Assignment Problem and Extensions. In Ding-Zhu Du and Panos M. Pardalos, editors, *Handbook of Combinatorial Optimization*, pages 259–311. Springer US, 2004.

[9] Fabio Parisi, Francesco Strino, Boaz Nadler, and Yuval Kluger. Ranking and combining multiple predictors without labeled data. *Proceedings of the National Academy of Sciences*, 111(4):1253–1258, January 2014.

[10] G. Terry Ross and Richard M. Soland. A branch and bound algorithm for the generalized assignment problem. *Mathematical Programming*, 8(1):91–103, December 1975.

[11] P. Sajda, E. Pohlmeyer, Jun Wang, L.C. Parra, C. Christoforou, J. Dmochowski, B. Hanna, C. Bahlmann, M.K. Singh, and Shih-Fu Chang. In a Blink of an Eye and a Switch of a Transistor: Cortically Coupled Computer Vision. *Proceedings of the IEEE*, 98(3):462–478, March 2010.