Deep Value of Information Estimators for Collaborative Human-Machine Information Gathering[*]

Kin Gwn Lore
Iowa State University
Ames, IA-50010, USA
kglore@iastate.edu

Nicholas Sweet
University of Colorado
Boulder, CO-80309
nisw6751@colorado.edu

Kundan Kumar
Iowa State University
Ames, IA-50010, USA
kkumar@iastate.edu

Nisar Ahmed
University of Colorado
Boulder, CO-80309
nisar.ahmed@colorado.edu

Soumik Sarkar
Iowa State University
Ames, IA-50010, USA
soumiks@iastate.edu

ABSTRACT

Effective human-machine collaboration can significantly improve many learning and planning strategies for information gathering via fusion of 'hard' and 'soft' data originating from machine and human sensors, respectively. However, gathering the most informative data from human sensors without task overloading remains a critical technical challenge. In this context, Value of Information (VOI) is a crucial decision-theoretic metric for scheduling interaction with human sensors. We present a new Deep Learning based VOI estimation framework that can be used to schedule collaborative human-machine sensing with efficient online inference and minimal policy hand-tuning. Supervised learning is used to train deep convolutional neural networks (CNNs) to extract hierarchical features from 'images' of belief spaces obtained via data fusion. These features can be associated with soft data query choices to reliably compute VOI for human interaction. The CNN framework is described in detail, and a performance comparison to a feature-based POMDP scheduling policy is provided. The practical feasibility of our method is also demonstrated on a mobile robotic search problem with language-based semantic human sensor inputs.

[*]This material is based upon work partially supported by the National Science Foundation under Grant No. CNS-1464279 and the NSF IUCRC Center for Unmanned Aerial Systems (CUAS).

1. INTRODUCTION

The notion of human-machine interaction for cooperative problem solving has attracted much interest in various cyber-physical system domains. Much of the human-machine interaction literature in the AI, robotics and controls communities tend to focus on the idea of humans acting in the role of collaborative planners or controllers for machine counterparts. In this work, however, we examine the problem of using humans as 'soft data sensors' for intelligent machine systems. In particular, we consider how large dictionaries of semantic human observations can be used to augment state estimates of measurable dynamical physical states that must be monitored continuously by conventional 'hard' sensor data used by machines (object position, velocity, attitude, temperature, size, mass, etc.).

Several modeling approaches have been developed to exploit soft human sensing across a variety of interfaces, e.g. verbally reported range and bearing for target localization [5]; verbal and sketch-based detection/no detection reports for target search [3, 1]; and semantic language inputs for target localization [2]. While these works have largely focused on developing human sensor models and suitable data fusion algorithms for blending hard and soft data, relatively little work has been done on *active soft sensing*, i.e., intelligent querying of human sensors to gather information that would be most beneficial for complex machine planning and/or perception tasks. Active sensing problems have a rich tradition in target tracking and controls communities, but have focused on hard data sources such as radar, lidar, cameras, etc. One particularly relevant issue is the *sensor scheduling problem*, i.e., the selection of most appropriate sensing assets given constraints on how many can be tasked to deliver data any given instant (e.g., due to bandwidth or computational limits). In this work, we address the problem of scheduling interactions between human sensors and their machine counterparts: what information should be solicited from human sensors to get most valuable soft information back out for machine counterparts, and when/how should such soft information be obtained? These issues can be tackled within formal planning frameworks that seek to maximize the *value of information* (VOI) under uncertainty, so that soft data is only provided by human sensors if it is worth the cost of using limited machine and human cognitive resources to obtain it. However, exact inference and optimization for VOI-based human-machine interaction is computationally expensive, and approximations for efficient online interaction must be sought.

This paper proposes a new deep learning based VOI estimation approach that aims to learn the policy reward given a joint state-action configuration. Deep learning is an emerging branch of machine learning that uses multiple levels of abstractions (from low-level features to higher-order representations, i.e., features of features) learnt from data without any hand-tuning. Among various deep learning tools, convolutional neural networks (CNNs) [7] are an attractive option for extracting pertinent hierarchical features from images for detection, classification, and prediction. In addition to this deep learning architecture, other architectures such as deep belief networks, deep autoencoders, and deep recurrent neural networks have also gained immense traction, as they have recently been shown to outperform all other state-of-the-art machine learning tools for handling very large dimensional data spaces. While many current applications involve image, video, speech and natural language processing, very recent studies have begun to explore applicability to decision and control problems such as reinforcement learning [13] and guided policy search [18, 11]. In this context, although supervised learning of a state to action policy map may be a straightforward formulation, we show that similar performance can be achieved while learning the reward (VOI in this case) given the state and action combination via use of a deeper architecture. We compare the performance of the deep learning based VOI estimator with a more traditional, hand-tuned policy learner such as an augmented Markov Decision process (AMDP). A feasibility study is presented with a realistic human-machine information gathering scenario for a dynamic search problem.

2. HARD/SOFT SENSOR FUSION

2.1 Problem Setup

For concreteness, we will mainly focus throughout this work on information-gathering tasks involving localization of moving intruders (targets) in large environments that are incompletely covered by networks of mobile/static human and machine sensors. This serves as a useful analog for large-scale intelligence, surveillance and reconnaissance applications [8]. For simplicity, we focus here on localizing a single known target in a 2D environment with a known map. We also focus here on a centralized sensor network architecture, in which all machine and human sensors report observations to a single processing point. We assume for now that a single human sensor is tasked to provide soft data observations and is always connected to the fusion center through an appropriate interface (e.g. a workstation console or mobile device).

At discrete time step k, let $X_k = X \in \mathbb{R}^2$ be the unknown target location with initial prior probability distribution $p(X_0)$ at $k = 0$. Assume that the environment can be modeled as a 2D grid with n_g total elements, so that X_k takes on discrete realizations x_k^i for $i \in \{1, ..., n_g\}$. A discrete time Markov chain with known transition probability $p(X_k|X_{k-1})$ is used to account for the target's uncertain motion through the environment. This distribution can capture a wide variety of possible target dynamics, including highly nonlinear behaviors with non-Gaussian process noise. The Chapman-Kolmogorov equation dictates the evolution of $p(X_{k-1})$ to $p(X_k)$,

$$p(X_k) = \sum_{x_{k-1}^i} p(X_k|X_{k-1} = x_{k-1}^i)p(X_{k-1} = x_{k-1}^i).$$

If hard sensor observations O_k^h and soft sensor observations O_k^s are available at each time step in the superset of observations $\mathcal{O}_k = \{O_k^h, O_k^s\}$, then the recursive Bayes' filter replaces $p(X_k)$ with the conditional posterior distribution $p(X_k|\mathcal{O}_{1:k})$, where $\mathcal{O}_{1:k} = \{\mathcal{O}_1, \cdots, \mathcal{O}_k\}$ and Bayes' rule gives

$$\begin{aligned}
&p(X_k|\mathcal{O}_{1:k}) \propto p(X_k|\mathcal{O}_{1:k-1})p(\mathcal{O}k|X_k),\\
&\propto p(X_k|\mathcal{O}_{1:k-1})p(O_k^h|X_k)p(O_k^s|X_k)\\
&\propto \sum_{x_{k-1}^i} p(X_k|X_{k-1})p(X_{k-1}|\mathcal{O}_{1:k-1})p(O_k^h|X_k)p(O_k^s|X_k),
\end{aligned}$$

where O_k^h and O_k^s are assumed conditionally independent given the true target state. The posterior summarizes all information gathered by all sensors up to time k and thus efficiently updates the belief in X_k without requiring storage of $\mathcal{O}_{1:k}$. The information from each sensor is encapsulated by the observation likelihood functions $p(O_k^h|X_k)$ and $p(O_k^s|X_k)$. For hard sensors, $p(O_k^h|X_k)$ is typically derived from hardware specifications, or parametrically modeled and estimated via calibration. For soft sensors, however, $p(O_k^h|X_k)$ must be approximated as a function of X_k depending on the type of human sensor interface used for a particular application. Conventional approximate Bayesian filtering techniques for non-Gaussian hard sensor data fusion (e.g. using grid, particle, or Gaussian mixture Kalman filter representations) can be naturally extended to incorporate fusion of soft data provided in various forms, including binary detection/no detection observations [1, 3] and semantic natural language observations [2, 14]. Suitable models $p(O^s_k|X_k)$ can be learned from data to properly account for uncertainty in human-generated information and capture key observation characteristics when deployed with real (expert or non-expert) humans. The sensor scheduling problem in this context thus be-

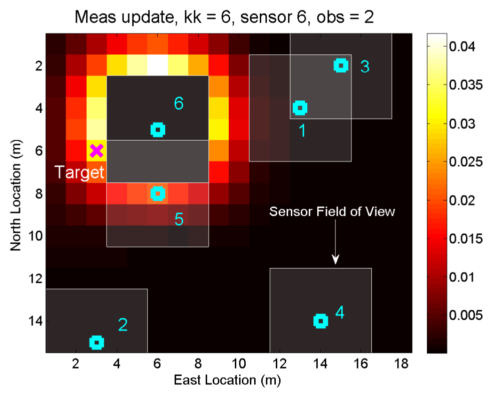

Figure 1: Simulation of 2D target localization problem, showing camera 1-6 locations and fields of views, posterior distribution $p(X_k|\mathcal{O}_{1:k})$ (heat map) and true target location (magenta x) for random walk model $p(X_{k+1}|X_k)$. Sensors have 0.99 detection rate and false alarm rate of 1×10^{-4}, 0.1504, 0.1585, 0.1992, 0.1510, and 0.1593, respectively.

comes one of obtaining an appropriate set of hard/soft measurements in the superset $\mathcal{O}_{1:k}$ so that the posterior distribution remains as informative as possible for constrained intelligent decision-making (e.g. deploying security assets to intercept the intruder within a certain time window). For example, consider the 2D grid world shown in Fig. 1, which features 6 cameras located in large open environment with a moving target that obeys random walk dynamics. Some cameras are linked to automatic target recognition (ATR) algorithms, with known true detection and false alarm rates that define $p(O^h_k|X_k)$. A remote human analyst can the remaining cameras to declare whether or not the target is in view (i.e. independently of ATR software). Due to varying quality of each camera and various cognitive loads, the analyst may incorrectly declare that the target has (not) been detected with some true detection and false alarm rates defining $p(O^s_k|X_k)$. The data fusion center can only request one hard observation O^s_k or soft observation O^h_k at any given k from a single camera, due to bandwidth and processing restrictions. The scheduling problem thus addresses the question of which single hard or soft sensor to query for the Bayes filter update at time k. This leads to the more general problem of obtaining an optimal sequence of hard/soft sensor queries to maximize some utility function over time. Matters become even more challenging when considering that human sensors can also provide detailed semantic observations, e.g., ‘Target is now headed north towards the door very quickly’. In this case, given a known dictionary of grounded semantic statements that can be directly translated into X_k information, the data fusion center could schedule and execute the most informative semantic queries over time, which the human sensor could answer in a binary ‘yes/no’ manner (‘Is the target moving to the door in camera 2?’).

2.2 VOI for Sensor Scheduling

Previous work on combined soft/hard sensing focused primarily on instances where human sensors voluntarily ‘push’ useful information as they see fit. However, this can lead to suboptimal gains for machine sensing and planning performance, especially if the analyst becomes distracted or fails to recognize when machine sensors are unable to collect good data. This leads us to consider formal strategies for opportunistically ‘pulling’ information from human sensors in the right way at the right time to enhance state estimation and long-term decision-making under uncertainty. Such interactions should also account for the costs of interacting with human sensors, in order to manage their limited cognitive resources and avoid overloading them with repetitive or marginally useful sensing tasks.

These issues are naturally addressed via formal decision-theoretic Bayesian inference to assess the *value of information* (VOI) for possible soft data queries [6]. For the simple target localization problem, suppose $O^s_k \in \left\{o^{s,1}_k, ..., o^{s,n_s}_k\right\}$, where $o^{s,j}_k \in [0, 1]$ denotes a specific kind of binary soft observation report. For instance, $o^{s,j}_k$ could represent a detection/no detection event for camera j, or a binary true/false response to a semantic query j from a large dictionary. Given utility function $U(D_k, X_k)$ for the expected long-term benefit of taking discrete action D_k while the target is in state X_k, the VOI for receiving a single noisy report $o^{s,j}_k$ in response to a soft data query is

$$\mathrm{VOI}(o^{s,j}_k) = \tag{1}$$
$$\mathbb{E}\left[\max_{D_k} U(D_k, X_k)\right]_{(o^{s,j}_k, X_k)} - \max_{D_k} \mathbb{E}\left[U(D_k, X_k)\right]_{(X_k)},$$

where $\mathbb{E}\left[f\right]_{(v)}$ is the expected value of f over random variables v, and

$$\mathbb{E}\left[\max_{D_k} U(D_k, X_k)\right]_{o^{s,j}_k, X_k} =$$

$$\sum_{o^{s,j}_k} p(o^{s,j}_k|\mathcal{O}_{1:k-1}) \left[\max_{D_k} \sum_{x^i_k} p(X_k|\mathcal{O}_{1:k-1}, o^{s,j}_k) U(D_k, X_k)\right],$$

$$\max_{D_k} \mathbb{E}\left[U(D_k, X_k)\right]_{X_k} = \max_{D_k} \sum_{x_k^i} p(X_k|\mathcal{O}_{1:k-1})U(D_k, X_k).$$

Assuming cost $c(o_k^{s,j})$, the human sensor should be queried for $o_k^{s,j}$ if $\text{VOI}(o_k^{s,j}) > c(o_k^{s,j})$. Thus, (1) gives a formal way to assess whether the expected information from $o_k^{s,j}$ is worth the cost of retrieving it, regardless of the outcome of $o_k^{s,j}$. The key idea in eq. (1) is to compare the maximum expected utility given all possible outcomes for X_k and $o_k^{s,j}$ to the maximum expected utility if no new soft data were obtained. In practice, we must compare the VOI at time k for n_s alternative sensor queries $o_k^{s,j}$, $j \in \{1, ..., n_s\}$, and select only the query with the highest VOI. This is referred to as a *myopic approximation*, since it does not consider all possible combinations of observations $o_k^{s,j}$ that could be taken together at time k. For dynamic X_k, this approach is also myopic since it does not consider future sensing actions for time $k+1$ and beyond. However, the definition of VOI can be generalized to find an optimal non-myopic soft *querying sequence* $O^s_{k:k+T}$ for $T > 0$, by assessing the single best query O^s_k at each time step $k, ..., k+T$, and comparing the final expected utility at step $k+T$ to the expected utility with respect to $p(X_{k+T}|X_k, \mathcal{O}_{1:k})$ (the propagated belief without any future soft observations).

VOI depends on $U(D_k, X_k)$ and $c(o_k^{s,j})$, as well as the uncertainty in $p(X_k|\mathcal{O}_{1:k})$. Here, $c(o_k^{s,j})$ can be related to the expected cognitive cost of tasking human sensors [6]. As there is no standard way to define $c(o_k^{s,j})$ for general applications, for simplicity and without loss of generality, we ignore $c(o_k^{s,j})$ for now and only consider utility defined by expected information gain. In this case, D_k is related only to the choice of $j \in \{1, ..., n_s\}$. We seek here to minimize the entropy of $p(X_k|\mathcal{O}_{1:k-1}, o_k^{s,j})$, so that

$$U(o_k^{s,j}, X_k) = \log p(X_k|\mathcal{O}_{1:k-1}, o_k^{s,j}). \qquad (2)$$

Hence, the VOI for $o_k^{s,j}$ in (1) becomes the expected decrease in posterior entropy,

$$\text{VOI}(o_k^{s,j}) = \mathbb{E}\left[\mathcal{H}[p(x_k|\mathcal{O}_{1:k-1}, o_k^{s,j})]\right]_{(o_k^{s,j})} - \mathcal{H}[p(x_k|\mathcal{O}_{1:k-1})],$$

where $\mathcal{H}[p(x_k|\mathcal{O}_{1:k})] = \mathbb{E}\left[\log p(x_k|\mathcal{O}_{1:k-1})\right]_{(x_k)}$

This means that soft data $o_k^{s,j}$ will be (myopically) requested to keep the overall spread of uncertainty in $p(x_k|\mathcal{O}_{1:k})$ as small as possible. Entropy minimization has been widely used for hard sensor tasking in dynamic target tracking [4], and thus provides a useful common objective for combined hard-soft sensor scheduling. Note, however, that other utility measures such as mutual information could also be used here instead [9].

2.2.1 *Practical VOI inference and optimization*

VOI-based scheduling requires NP-hard Bayesian inference calculations for the marginal observation likelihoods $p(o_k^{s,j}|\mathcal{O}_{1:k-1})$. Comparison of VOI for various $o_k^{s,j}$ reports can be quite expensive when n_s is large (i.e. for large semantic dictionaries), or if temporally non-myopic query sequences over multiple time steps are considered (due to combinatorial blow up). Hence, even for myopic approximations, it is generally impractical to explicitly assess VOI *online* among n_s soft sensing alternatives. Many approximate inference methods have been developed to address these issues [12], but still rely on approximate online inference and expensive online optimization.

VOI-based sensor scheduling problems can also be interpreted as partially observable Markov decision processes (POMDPs) [10], which allow *optimal sensing policies* $\pi(b(X_{k-1}))$ to be computed *offline* via value iteration over the belief space $b(X_{k-1}) = p(X_k|\mathcal{O}_{1:k-1})$. Such policies encode 'look-up tables' from $b(X_{k-1})$ to queries $o_k^{s,j}$ that automatically account for VOI through expected cumulative rewards, and thus allow for efficient online optimal querying that bypasses explicit comparison of all possible future sensing actions. As discussed in Section 4, POMDP approximations such as augmented Markov decision processes (AMDPs) [15, 17] can be used to derive sensing policies by first learning and then solving MDPs over *belief space features* $f(b(X_k))$. AMDPs recover the linear additive reward structures in $f(b(X_k))$ space and thus enable the use of standard value iteration solutions for offline policy generation in non-standard POMDPs. However, the corresponding MDP model parameters must be obtained via simulation for offline policy calculation, subject to hand-tuning of discount and immediate expected reward parameters, as well as proper selection of features $f(b(X_k))$.

Nevertheless, the AMDP approximation for POMDPs provides an important insight into another general approximation strategy for VOI-based scheduling: optimal querying policies could be obtained offline via supervised learning, using features of $b(X_k)$ and simulated observation instances to provide generalizable VOI associations between $b(X_k)$ and O^s_k queries. The learned policy could be trained for non-myopic querying, and would thus provide a computationally efficient means for pulling information from soft human sensors that avoids expensive online brute force VOI optimization over all possible semantic query sequences.

3. DEEP VOI ESTIMATORS

Deep neural networks present such an attractive option for estimating VOI without policy hand-tuning. Before describing the deep VOI estimation framework, we begin this section a brief background on deep Convolutional Neural Networks (CNN).

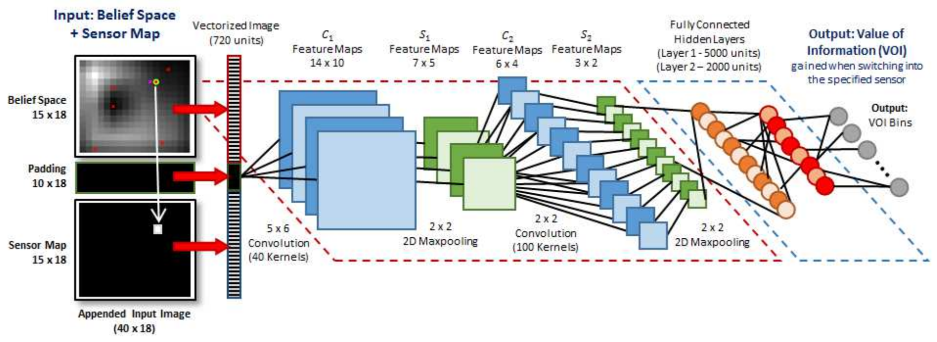
Input: Belief Space + Sensor Map
Belief Space 15 x 18
Padding 10 x 18
Sensor Map 15 x 18
Appended Input Image (40 x 18)
Vectorized Image (720 units)
C_1 Feature Maps 14 x 10
S_1 Feature Maps 7 x 5
C_2 Feature Maps 6 x 4
S_2 Feature Maps 3 x 2
Fully Connected Hidden Layers (Layer 1 - 5000 units) (Layer 2 – 2000 units)
Output: Value of Information (VOI) gained when switching into the specified sensor
Output: VOI Bins
5 x 6 Convolution (40 Kernels)
2 x 2 2D Maxpooling
2 x 2 Convolution (100 Kernels)
2 x 2 2D Maxpooling

Figure 2: **Schematic of the CNN used for reward learning.**

3.1 Deep Convolutional Networks

Deep CNN is a suitable choice due to its ease of training. Compared with fully connected networks with the same number of hidden layers, it is able to achieve a similar performance with fewer parameters to learn. The main idea here is to use CNN visual image processing techniques to learn the mapping from from the belief vector $b(X_k)$ to the optimum set of sensing actions. The 'pixels' of the 2D visual image are simply the elements of the $b(X_k)$ vector. CNNs are designed to exploit the 2D structure of an input image by preserving the locality of features via the utilization of spatially-local correlations of an image through the use of tied weights (shown in Fig. 3), therefore being invariant to translation of feature positions. Weight-sharing increases efficiency of learning because the number of free parameters being learned while training are greatly reduced as opposed to a fully connected neural network.

In CNNs, data is represented by multiple feature maps in each hidden layer. Feature maps are obtained by convolving the input image by multiple filters in the corresponding hidden layer. In other words, they are obtained by repeatedly applying a function across subregion over the entire image, i.e., a convolution operation of the input image with a linear filter.

To further reduce the dimension of the data, these feature maps typically undergo non-linear downsampling with a 2×2 or 3×3 maxpooling. Maxpooling partitions the input image into sets of non-overlapping rectangles and takes the maximum value for each partition as the output. As neighboring pixels in an image share similar features, these pixels can be discarded with the benefit of overcoming memory constraints and decreasing training time. Furthermore, both spatial and feature abstractness can be increased by maxpooling. This results in increased position invariance of the filters. After maxpooling, multiple dimension-reduced vector representations of the input are acquired and the process is repeated in the next layer to learn a higher representation of the data. At the final pooling layer, resultant outputs are linked with the fully connected layer where sigmoid outputs from the hidden units are joined with output units to infer a predicted class based on the highest joint probability given the input data.

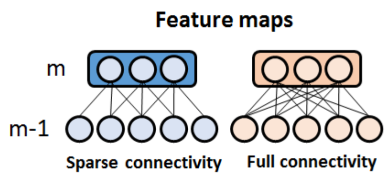

Figure 3: **Sparse connectivity between layers m and $m-1$ compared with full connectivity in a fully connected network.**

The probability of an input vector $\mathbf{v}$ being a member of the class i can be written as:

$$\Pr(Y=i|\mathbf{v},\mathbf{W},\mathbf{b}) = \mathrm{softmax}_i(\mathbf{W}\mathbf{v}+\mathbf{b}) = \frac{e^{\mathbf{W}_i\mathbf{v}+\mathbf{b}_i}}{\sum_j e^{\mathbf{W}_j\mathbf{v}+\mathbf{b}_j}} \tag{3}$$

and the prediction of the model is the class with the highest probability:

$$y_{\mathrm{pred}} = \mathrm{argmax}_i \Pr(Y=i|\mathbf{v},\mathbf{W},\mathbf{b}) \tag{4}$$

The model weights, $\mathbf{W}$ and biases, $\mathbf{b}$ are optimized by an error backpropagation algorithm, where true class labels are compared against the model prediction using an error metric that becomes the loss function of the algorithm. Specifically, the loss function ℓ to be minimized for a dataset $\mathbf{V}$, parametrized by θ is:

$$\ell(\theta=\{\mathbf{W},\mathbf{b}\},\mathbf{V}) = -\sum_{i=0}^{|\mathbf{V}|}\left[\log\left(\Pr(Y=y^{(i)}|v^{(i)},\mathbf{W},\mathbf{b})\right)\right] \tag{5}$$

where $y^{(i)}$ denotes the class index.

3.2 Design of framework

Policy Learning: A typical formulation of the problem is directly mapping the belief space to the action by using belief maps as inputs and sensor indices as outputs. For this problem set-up, a single belief map is accompanied by an index denoting the action of switching into the sensor that gives the best VOI gain. The belief map is represented by a $n_y \times n_x$ (rows $\times$ columns, in pixels) grayscale image, where pixels with high grayscale intensity denote high belief and pixels with low grayscale intensity denote low belief. The belief map is then vectorized into a vector of $1 \times n_g$ where $n_g = n_x n_y$ (from Section 2) and goes through a typical convolutional neural network to make a prediction that suggests which sensor to switch into for the most VOI gain.

Reward Learning: Another formulation of the problem that is perhaps more robust, we aim to learn the underlying function for predicting VOIs given a belief map and an arbitrary sensor index. For this formulation, the CNN architecture is similar but the inputs are modified to include an arbitrary sensor. The Cartesian coordinates of this sensor is indicated on a map with the same dimensions of the belief map called the *sensor map*. In addition, enough padding is added between the belief map and the sensor map such that the information from these two spaces do not interfere with the learning of filter weights. The belief map $(n_y \times n_x)$, padding $(n_p \times n_x)$, and sensor map $(n_y \times n_x)$ are vertically concatenated to form a $(2n_y + n_p) \times n_x$ image that will become the input to the CNN. VOI also becomes the true class labels associated with this input image during training. However, since VOI is real-valued, we have discretized the VOI space into n_c classes to formulate a classification problem. Hence, the outputs of the model (i.e., the classes) correspond to the estimated VOI given a sensor and a belief map. Fig. 2 shows a schematic of this formulation.

Using Eq. (3), (4) and (5), the VOI can be estimated based on the associated action of switching into a particular sensor given the current belief map.

Training Data Generation: Data is generated from running the simulation for 10,000 time steps with random sensor false alarm rate of $15 \pm 5\%$ and the target moving in a random walk manner. Depending on the formulation of the problem, the resultant size of the dataset varies. In the policy learning formulation, we are limited to one training example per time step since there is only one optimal sensor index given a single belief map. This results in 10,000 examples, where half of them (i.e., 5,000) are used for training and the other half is used for cross-validation to avoid overfitting. The reward learning formulation allows us to produce six examples per time step (since there are six sensors in the field) and generated a total of 60,000 examples. In this case, we consider the VOI gained 4 time steps ahead in future, in a non-myopic fashion. The expected VOI gained by switching into the particular sensor is used as the class labels after discretization. The number of classes n_c is 62 (the VOI spans between 0.15 and 0.46, so the interval size for a class is 0.005). Again, half of the example data is used for training and the other half is used for validation.

Network Architecture and Hyper-parameters: Specific details on the input dimensions, number of filters, convolutional layer, pooling layers, and fully connected layers are shown in and Fig. 2. A learning rate (also known as step size) of 0.01 is used for the gradient descent algorithm for training the CNN in a supervised manner with a batch size of 10 samples. The optimized model is acquired prior to the point when validation error becomes consistently higher than the training error in subsequent training iterations.

4. RESULTS AND DISCUSSION

In this section, the performance of CNN model is evaluated and compared with AMDP.

4.1 Prediction Accuracy

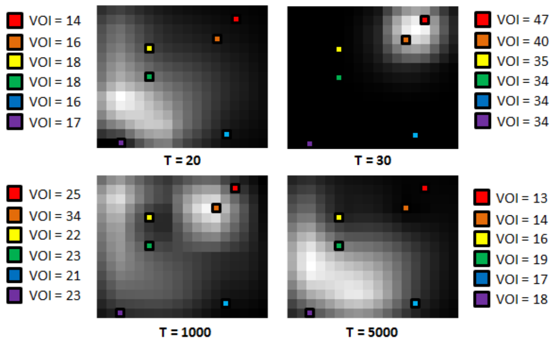

Figure 4: Four examples of belief map and VOI (discretized into bins) associated with the action of switching into a particular sensor at different simulation time steps T.

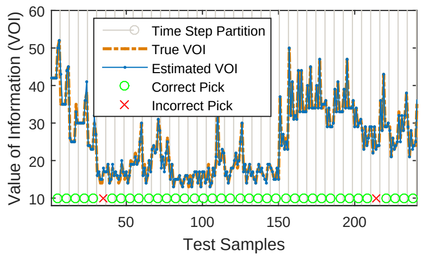

Figure 5: Estimated VOI vs. True VOI for the first 40 out of 10,000 simulation time steps. The figure also shows whether the sensor selected through rank-ordering the predicted VOI is the same as the sensor selected by ordering the true VOI, represented by green circles (matching) and red crosses (non-matching).

The VOI estimated by the trained model follows closely to the true VOI computed as shown in Fig. 5. Each point in one time step partition represents a sensor index; the model has to select one sensor out of six based on the current belief map with the highest expected VOI gain. A successful selection is represented by a green circle, whereas a wrong selection is represented by a red cross. It is observed that CNN can be in fact trained to learn the reward function and produce VOIs similar to the values computed by the brute force simulation. Most importantly, the high capability of predicting this non-myopic VOI suggests that the model does not naively generate VOI that is directly proportional to the grayscale pixel intensity of the belief map (which corresponds to the probability of finding the target) at that particular sensor location. Furthermore, careful inspection of the camera choice errors reveal that most of the errors come from confusing between camera 5 and camera 6 that are spatially very close as well as tend to have similar VOI reward for many belief space configurations.

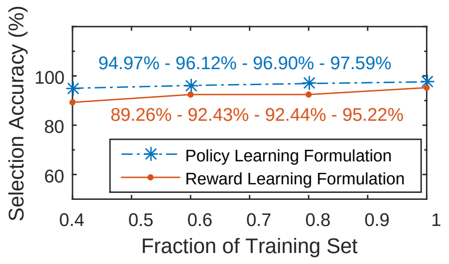

Figure 6: **Degradation of selection accuracy with reduction of dataset sizes used to train the CNN model.**

Results also suggest that the trained CNN model is able to generalize into unseen belief maps and still predict VOI that is close to the true VOI. To show this, we have reduced the size of the dataset used to train the CNN model with $p \in \{0.4, 0.6, 0.8, 1.0\}$ denoting the fraction of the new training and validation set sizes from their original sizes. From Fig. 6, the accuracy of selecting the best sensor with largest VOI gain decreases slightly with smaller dataset sizes. Policy learning formulation (i.e., direct action mapping) seems to perform slightly better compared to reward learning (i.e., mapping into VOI) as it suffers only a slight degradation in prediction performance. Furthermore, policy learning is attractive because it is straightforward and intuitive - given a belief map, a decision can be made to choose the best sensor. Note, the reward learning framework needs a deeper architecture (i.e., an extra fully connected layer) to achieve a similar performance compared to the policy learning framework. Training CNN for policy is also a little easier as it allows a simpler model with the number of output classes equivalent to the number of available options (i.e., choosing a particular sensor). On the other hand, the ability to evaluate the VOI from an action carried out in a specific belief space is very useful in situations where the location/charecteristics of the sensors may be altered slightly. Additionally, the framework can be more flexible in terms of additional objectives or imposing extra constraints (e.g., the predicted VOI can be processed to include penalty terms for certain actions). If this happens, the policy learning model must be retrained whereas the reward learning model may not need to be retrained. The adaptability of the reward learning formulation is absolutely valuable for generalizing into larger problem setups by paying a small price in accuracy.

4.2 Comparison with AMDP

A policy derived from a feature-based solution to the POMDP model for the sensor scheduling problem can be used to provide a baseline comparison with the learned CNN policies. As mentioned in Section 2, augmented Markov Decision processes (AMDPs) can be used to solve the non-standard POMDP for sensor scheduling when the reward function is defined to minimize the entropy of the posterior state belief $b_k = p(X_k|\mathcal{O}_{1:k})$. AMDPs use a learned Markov decision process (MDP) model on features $f(b_k)$ of the belief space, which can include the entropy of b_k. This MDP can then be used to derive scheduling policies via offline value iteration with linear additive rewards defined in terms of $f(b_k)$ instead of X_k. By choosing a good set of features, the optimal expected total reward for the AMDP policy can closely match that of the original POMDP defined over X_k with non-standard entropy rewards. As discussed in [15, 17], the key idea behind the AMDP is that most of the reachable belief space b_k evolves along a low-dimensional manifold, which can be encoded by a feature set whose size is typically much smaller than the number of possible states X_k in the original POMDP. Hence, AMDPs offer a generative feature-based alternative to finding scheduling policies, in contrast to the discriminative feature-based modeling approach used by CNNs.

However, the problem remains to find suitable features $f(b_k)$ and learn the corresponding MDP model over the feature space, which is a non-trivial learning problem. Furthermore, discount and immediate expected reward parameters for the AMDP must be hand-tuned to arrive at suitable policies via value iteration. Finally, stationary infinite horizon policies must often be used in practice to avoid the computational cost of performing finite time value iterations for non-stationary scheduling policies (especially if discount and reward parameters must be tuned). The use of infinite horizon policies necessarily makes the AMDP suboptimal for finite-horizon querying problems, but nevertheless provides insight into the expected capabilities of generative feature-based learning approaches for sensor scheduling policy estimation. Our implementation of AMDP for the simple 2D grid world problem follows the basic technique presented in [17], which defines $f(b_k)$ as the stacked vector of the maximum a posteriori (MAP) state X_k of $b(k)$ and the (discretized) entropy of $b(k)$. The resulting implementation used 27,000 AMDP feature states, and the policy was generated using hand-tuned rewards (with positive rewards proportional to inverse square of entropy for transitions to lower-entropy states, and negative rewards otherwise) and a fixed value function discount factor of 0.1 (to encourage earlier transitions to low-entropy states).

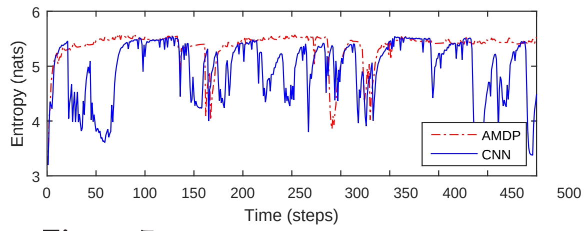

Figure 7: Entropy of belief map over time.

To study the level of certainty where the target is in a particular region, the entropy of the belief map at each time step is studied. High entropy means there is a large uncertainty that the target is in a given region, and low entropy suggests an ability of narrowing down the target location. The entropy of the belief map generated using AMDP and CNN is computed and plotted over 500 time steps, shown in Fig. 7. Belief entropy from AMDP experiences sharp dips in magnitude occasionally but stays high at all other times. Entropy from CNN is generally lower at most time steps, thus implying lower uncertainty. However, this may not be so simple to say, especially given the target's propensity to move around a lot–even if the model starts off knowing exactly where the target is, the entropy jumps up a lot at the next time step. Therefore, along with this decision uncertainty metric, a correctness metric is also used to compare the performances of AMDP and CNN.

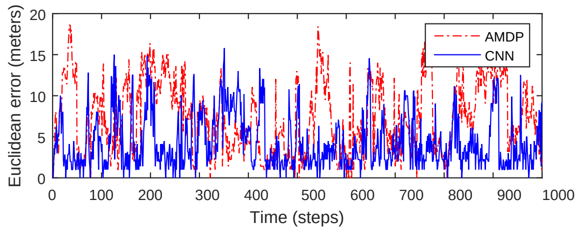

Figure 8: Euclidean error between MAP estimate and target location over time.

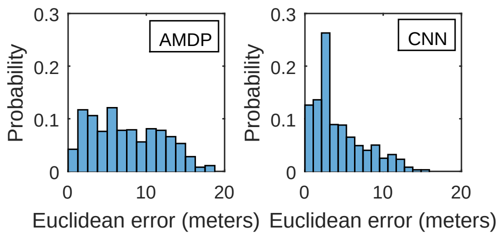

Figure 9: Error distribution between MAP estimate and target location.

Estimation error is defined as the Euclidean distance between the maximum a posteriori probability (MAP) estimate (i.e., largest value of the belief map) and the actual target location. Ideally, this error should be as close to zero as possible. The error is calculated using belief map and target locations for each time step (see Fig. 8) and its distribution is shown in Fig. 9. Clearly, CNN outperforms AMDP as the distribution is more skewed with higher probability of the belief map mode being closer to the actual target location, hence increasing the success rate of target-tracking.

As a hierarchical feature extraction tool, deep CNN is able to extract belief space features at different spatial scales as well as obtain 'features of features'. Also, belief spaces generally evolve on low-dimensional manifolds (i.e., feature spaces) as suggested by the AMDP formulation. Therefore, it becomes feasible for a deep CNN to associate these representative features to the VOI. In this context, we make an interesting observation (as shown in Fig. 6) that CNN performance does not degrade much with a drastic decrease in the training data size. This probably suggests that there is a relatively low number of key features in the belief space that (the CNN is able to learn) lead to various belief space configurations via complex combinations. Another view could be that the proposed framework aims to automatically learn the probability density of the reward given belief space and action. Therefore, with a hierarchy of nonlinear functions and a large number of model parameters, it becomes feasible to model arbitrarily complex densities. However, sufficient training data and regularization steps are necessary to avoid overfitting.

5. SEMANTIC SOFT DATA SCHEDULING

Consider a 'cops and robbers' scenario in which one robot (the 'cop') searches for another mobile intruder robot (the 'robber') in an indoor environment, with a simulated remote human 'security guard' providing observations of the robber's 2-dimensional position state. As in the previous grid world toy problem, the cop maintains a Bayesian belief map over the environment of the robber's state, and updates this belief either through fusion of hard sensor measurements (e.g. detection of robber position via an on-board camera) or soft data. The soft data in this case takes the form of a human semantic observation as in [2], e.g. "The robber is in front of the desk." We assume that the remote human views the scene either through the cop robot's camera, or a security camera placed in each room. The human thus has full visibility into the space, but is constrained to visibility of no more than one room per time step. Similarly, we have constrained the cop robot to asking a single question about the robber once every $n_q = 5$ time steps, as a way to mitigate operator load. Over time, the probability mass of the target position diffuses over the environment, based on the cop's estimate of the robber's position and known random-walk dynamics model.

With this setup, the human observations $o_k^{s,j}$ can be thought as binary 'true/false' responses to the questions asked by the cop robot, where j indexes an element from a finite list of n_s semantic questions. We assume $o_k^{s,j}$ arrive only as responses to questions posed by the robot, i.e. the human does not 'push' information voluntarily, and only one semantic query j is selected from $\{1, ..., n_s\}$. The cop can ask the human whether the robber is either inside one of the rooms or near one of the objects shown in Fig. 10. This leads to $n_s = 16$ possible questions that generate measurement updates. For example, if the cop asks, "Is the robber near the dining table?", a positive response triggers fusion of the likelihood shown in Fig. 11 with the current belief state. At any given query instance, the cop seeks to ask the question with maximum VOI, as defined by the expected reduction in entropy. The 16^T queries of depth T are ranked by their corresponding VOI; in the non-myopic case of $T \geq 2$, all question paths are ranked by VOI first and then reduced to the initial 16 possible questions, ranked by the maximum VOI of all query sequences starting with that question.

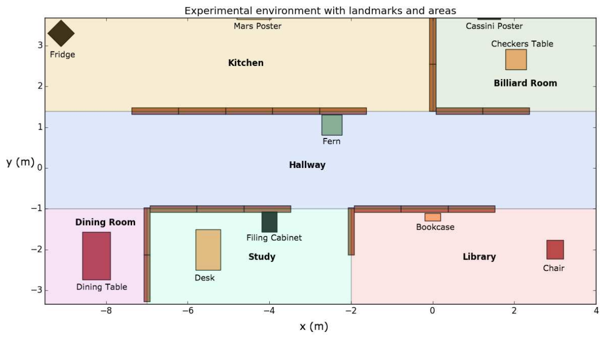

Figure 10: The indoor search environment and associated semantic features for soft observations.

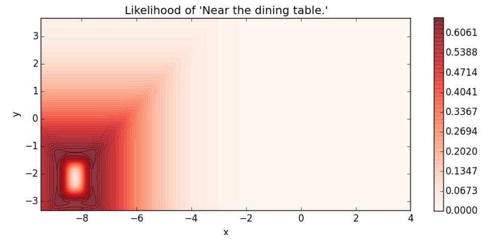

Figure 11: The likelihood function that maps to the example observation, 'Target is near the dining table'.

CNN training: This scenario provides a truth model used to train the CNN. Applying the same CNN input-output structure and problem formulation, the belief map (72×136) is appended with padding (10×136) and the action map (8×136) where the action map is divided into 16 equally spaced nodes that light up depending on the question posed by the cop. Enough padding is created to separate the belief map from the action map for efficient learning of the convolving filters. Fig. 12 illustrates the images used in learning. We trained the CNN model (only for myopic scenario for this feasibility study) with 31,680 training examples and 31,680 validation examples sampled from 100 simulation trials with 400 time steps each. Therefore, variability in the belief space is sufficient. As the input image size is now larger relative to the simpler problem, filter sizes are increased to 10 of 7×9 in the first convolutional layer and 25 of 5×5 in the second convolutional layer with all other parameters equal. Note that the length of the smaller padding edge (i.e., 10 pixels) is still larger than the longest filter dimensions.

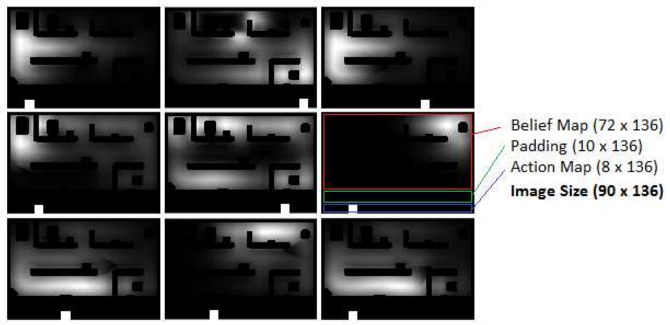

Figure 12: Nine examples of CNN input showing the belief map, padding, and the action map in the realistic scenario.

Results: From our simulations, the CNN is **91.33%** accurate in determining the next best question with highest VOI gain that should be posed by the robot (based on 1,980 time steps), as compared with the ground truth determined by brute force calculation. Interestingly, most of the errors occur as the target moves from one room to a neighboring one (e.g., from library to study) and the query selection sometimes remain associated with the previous room. The CNN-based 'query' selection quickly adjusts within a few time steps after such a transition. Hence, most of these errors can potentially be avoided with a non-myopic VOI implementation for this real-life scenario and is currently being pursued. Overall, this exercise demonstrates the feasibility of using deep learning architectures for such decision-making processes.

6. CONCLUSIONS AND FUTURE WORK

VOI-based human-machine interfaces can automatically determine how and when to present queries to human operators in a real problem. However, practical online implementation of VOI-based querying strategies remains challenging, since the problem of selecting the optimal sequence of queries leads to a difficult analytically intractable joint optimization and inference problem. This paper uses recent advancements in deep learning to build a VOI estimation framework that is shown to be able to reliably estimate VOI without any policy hand-tuning. A 2-D grid world search problem (with a moving target) is used to compare the performance of the finite horizon deep VOI estimator with that of a hand-tuned AMDP policy. Simulation results show that a CNN-in-the-loop information gathering system is able to lower the expected entropy of belief spaces and the expected error between the MAP estimate of the target and the true target compared to an AMDP-in-the-loop process. Finally, a feasibility study was performed on a simulation test bed with a realistic human-machine collaboration problem.

Better network design and hyper-parameter optimization are currently being investigated. Other future research directions are: (i) online tuning of deep networks; (ii) hybrid approaches involving deep feature extractors and traditional planning algorithms (e.g. using CNNs to learn feature maps for AMDP-based policy approximations); (iii) addressing sensor modeling issues, including potentially unknown false alarms/missed detection rates and imperfect human sensor models (e.g. using hierarchical Bayesian modeling as in [1] to account for model uncertainties); and (iv) validation of the proposed methodology on indoor robotic target search test bed with live human users. We will also extend the comparisons made here between feature-based direct policy learning approaches and feature-based POMDP approximations to other state-of-the-art POMDP approximations, including those that approximate low-dimensional reachable belief spaces via online sampling rather than through offline-learned feature compression [16].

7. REFERENCES

[1] N. Ahmed, M. Campbell, D. Casbeer, Y. Cao, and D. Kingston. Fully bayesian learning and spatial reasoning with flexible human sensor networks. In *Proceedings of the ACM/IEEE Sixth Int'l Conf. on Cyber-Physical Systems*, pages 80–89. ACM, 2015.

[2] N. Ahmed, E. Sample, and M. Campbell. Bayesian multicategorical soft data fusion for human robot collaboration. *IEEE Trans. on Robotics*, 29:189–206, 2013.

[3] F. Bourgault, A. Chokshi, J. Wang, D. Shah, J. Schoenberg, R. Iyer, F. Cedano, and M. Campbell. Scalable Bayesian human robot cooperation in mobile sensor networks. In *Int'l Conf. on Intelligent Robots and Sys.*, pages 2342–2349, 2008.

[4] M. F. Huber, T. Bailey, H. Durrant-Whyte, and U. D. Hanebeck. On entropy approximation for Gaussian mixture random vectors. In *2008 IEEE Int'l Conf. on Multisensor Fusion and Integration*, pages 181–188, Aug. 2008.

[5] T. Kaupp, B. Douillard, F. Ramos, A. Makarenko, and B. Upcroft. Shared environment representation for a human robot team performing information fusion. *Journal of Field Robotics*, 24(11):911–942, 2007.

[6] T. Kaupp, A. Makarenko, and H. Durrant-Whyte. Human robot communication for collaborative decision making: A probabilistic approach. *Robotics and Autonomous Systems*, 58(5):444–456, May 2010.

[7] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 2010.

[8] D. Kingston. Intruder Tracking Using UAV Teams and Ground Sensor Networks. In *German Aviation and Aerospace Congress (DLRK 2012)*, Berlin, Germany, 2012. German Society for Aeronautics and Astronautics (DGLR).

[9] A. Krause and C. E. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, 2005.

[10] V. Krishnamurthy and D. V. Djonin. Structured threshold policies for dynamic sensor scheduling: A partially observed markov decision process approach. *IEEE Transactions on Signal Processing*, 55(10):4938–4957, Oct. 2007.

[11] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *arXiv preprint arXiv:1504.00702*, 2015.

[12] Q. Liu and A. Ihler. Belief Propagation for Structured Decision Making. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2012.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 02 2015.

[14] B. Park, A. Johannson, and D. Nicholson. Crowdsourcing soft data for improved urban situation assessment. In *Int'l Conf. on Information Fusion (FUSION)*, pages 669–675. IEEE, 2013.

[15] N. Roy, G. Gordon, and S. Thrun. Finding Approximate POMDP Solutions Through Belief Compression. *Journal of Machine Learning Research*, 23:1–40, 2005.

[16] D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems*, pages 1–9, 2010.

[17] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005.

[18] T. Zhang, G. Kahn, S. Levine, and P. Abbeel. Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. *arXiv preprint arXiv:1509.06791*, 2015.