Toward Autonomous Mapping and Exploration for Mobile Robots through Deep Supervised Learning

Shi Bai, Fanfei Chen and Brendan Englot

Abstract—We consider an autonomous mapping and exploration problem in which a range-sensing mobile robot is guided by an information-based controller through an a priori unknown environment, choosing to collect its next measurement at the location estimated to yield the maximum information gain within its current field of view. We propose a novel and time-efficient approach to predict the most informative sensing action using a deep neural network. After training the deep neural network on a series of thousands of randomly-generated “dungeon maps”, the predicted optimal sensing action can be computed in constant time, with prospects for appealing scalability in the testing phase to higher dimensional systems. We evaluated the performance of deep neural networks on the autonomous exploration of two-dimensional workspaces, comparing several different neural networks that were selected due to their success in recent ImageNet challenges. Our computational results demonstrate that the proposed method provides high efficiency as well as accuracy in selecting informative sensing actions that support autonomous mobile robot exploration.

I. INTRODUCTION

We consider a mobile robot that has no prior knowledge of the contents of its environment and must make sequential decisions about where to travel next, comprising an autonomous exploration problem [1]. Specifically, we formulate an information-theoretic exploration problem in which the long-term goal is to reduce entropy throughout the robot’s environment map, and the short-term goal is to perform the sensing action in each iteration that will maximize mutual information (MI), along the lines of [2].

We assume the robot is equipped with a range sensor and uses an occupancy grid [3] to represent and reason about the environment. Our solution to this problem is motivated by the recent work of [4], [5] and [6], in which the first work proved that by choosing sensing actions which maximize MI, a robot will be driven to unexplored space in the limit, and the latter works showed that supervised learning can be used to predict informative actions without evaluating the expected MI exhaustively for every possible action. It has also been shown in [7] that planning to maximize MI is highly effective in large-scale 3D exploration tasks in practice.

However, the information theoretic approaches [5], [6] and [7] require a computationally intensive real-time evaluation of a set of candidate sensing actions, and this set may grow significantly when a robot has many degrees of freedom, and must map large-scale 3D environments. In this work, we propose and evaluate a methodology to choose sensing actions for exploration using a deep neural network, with the aim of selecting optimal or near-optimal sensing actions by utilizing the structure of the local sub-map surrounding the robot (Fig. 1), and avoiding the intensive ray casting required to explicitly compute mutual information in real-time.

A. Related Work

Among the earliest information-theoretic exploration strategies are those proposed by Whaite and Ferrie [8] and Elfes [2]. The former work proposes exploring an a priori unknown environment with the goal of minimizing entropy, and the latter work specifically proposes exploring to maximize the mutual information between sensor observations and an occupancy grid map. More recent works in information-theoretic exploration have considered the trade-off between maximizing mutual information and managing the localization uncertainty in a robot’s simultaneous localization and mapping (SLAM) process [9], [10], [11], in addition to the selection of trajectories that maximize map accuracy [12]. Efforts to reduce the computational cost of evaluating mutual information over many possible future measurements have considered small, carefully selected sets of candidate trajectories, using a skeletonization of the known occupancy map [13] and the evaluation of information gain over a finite number of motion primitives [7], [14] or 3D viewpoints [15]. Limiting consideration to local neighborhoods of configurations permits efficient exploration by manipulators in 3D environments [16].

Deep neural networks have been successfully applied to challenging problems such as image recognition [17], robot manipulation [18] and control [19], [20]. Recent work on obstacle avoidance [21] successfully trained a deep neural network which took RGB images as input and generated steering angles as output, while a robot was moving forward at constant speed. A novel approach for visual navigation proposed in [22] also took RGB images as input, and the learned model was able to recognize the cues for navigation to a target for which only the target’s appearance was
exposed to the network. As it is helpful to have the rich features available from RGB images, fewer works on deep neural networks have focused on range sensors. However, of those which have, one recent work learned to detect vehicles from LIDAR scans [23], and another learned to predict exit locations from building floorplan maps [24].

B. Paper Organization

A formal definition of the problem to be addressed is given in Section II, including details about how we select training samples and assign labels. In Section III we discuss the different neural networks investigated, and provide the details of the training process. Results given in Section IV consist of two parts: a) accuracy of one-step decision-making, and b) performance while applying the neural network to a full-length exploration task, with conclusions in Section V.

II. TECHNICAL APPROACH

A. Problem Formulation

Instead of choosing the sensing action of maximum expected information gain after a computationally expensive evaluation of all the candidate sensing actions, we train a neural network to predict the optimal sensing action using a database of maps with similar characteristics but different topology and layout. A flowchart demonstrating both the training and testing phases is shown in Figure 2. We train and test the neural network on locally visible portions of randomly-generated 2D maps of indoor environments, anticipated to be useful in cases when a locally visible portion of the map provides a cue to an informative sensing action. Every training sample of an exploration-in-progress is labeled with the sensing action of maximum expected information gain, and the trained model is tested over different map examples that are withheld from the training data set.

As illustrated in Figure 2, images containing a randomly-seeded robot's locally visible portion of a randomly generated 2D map are repeatedly sampled and set aside for either training or testing. For each image, the label is an index corresponding to the expected optimal action that allows the robot to collect the maximum amount of new information from its sensor. We consider these locally visible, uniformly-sized sub-maps instead of the entirety of a robot's currently acquired map with the aim of achieving a scalable, computationally efficient approach that may be applied to environments of arbitrary size. However, this is done with the understanding that we are training an information-seeking controller rather than a global planning method.

As a simulated robot explores each member of a set of pre-generated 2D maps, its labeling process is driven by a similar approach to that of [4] and [7] to select the anticipated optimal action at every step. More details about how the optimal actions were selected are described below in Section II-B. As in [4], [5], and [6], we train the robot to select only one sensing action at a time. The implications of extending the proposed method to systems with higher-dimensional state and action spaces will be discussed in Section V.

B. Label Calculation

We define Shannon's entropy [25] over an occupancy grid map $m$ as follows:

$$H(m) = -\sum_{i} \sum_{j} p(m_{i,j}) \log p(m_{i,j}),$$

where index $i$ refers to the individual grid cells of the map and index $j$ refers to the possible outcomes of the Bernoulli random variable that represents each grid cell, which is either
free or occupied. Cells whose contents have never been observed are characterized as \( p(m_{ij}) = 0.5 \), contributing one unit of entropy per cell. Cells whose contents are perfectly known contribute no entropy to the summation.

We use mutual information \( I(m, x_i) \) to evaluate the expected information gain with respect to a specific configuration \( x_i \), defined as follows:

\[
I(m, x_i) = H(m) - H(m|x_i), \tag{2}
\]

where \( H(m) \) is the current entropy of the map, and \( H(m|x_i) \) is the expected entropy of the map given a new (predicted) sensor observation at configuration \( x_i \). Our goal is to select the optimal robot configuration \( x^* \) whose sensor observations will maximize the expected information gain:

\[
x^* = \underset{x_i \in \mathcal{C}_{\text{action}}}{\text{argmax}} I(m, x_i). \tag{3}
\]

In Equation 3, \( \mathcal{C}_{\text{action}} \) represents the subset of the configuration space from which the robot’s next sensing action will be selected, which in our case is comprised of actions at a designated step distance from the robot’s current location, spanning the full range of feasible heading angles. The robot is moved to each of these prospective locations, its sensor rays are projected into the environment, and the expected information gain of each action is tallied, assuming that unobserved space is unoccupied, and rays will pass through unless a previously-observed obstacle presents an occlusion.

III. EXPERIMENTS WITH DIFFERENT NEURAL NETWORKS

A. Neural Networks

Deep neural networks [26] have been successfully applied to image classification problems such as the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [17] in recent years. We have selected deep neural network image classification methodologies representing mature frameworks that have been extensively tested and characterized, noting that the development of a novel structure for a deep neural network is non-trivial and out of the scope of this particular study. Below is a list of the different neural networks tested with our range-sensing robot exploration dataset:

1) AlexNet: AlexNet [27] was the pioneering winner of the 2012 ILSVRC, after which deep neural networks have repeatedly won the ILSVRC competition.

2) VGG: Though VGG [28] was not the winner of the 2014 ILSVRC, it is well known for the use of a much smaller filter size compared to AlexNet, although the network contains many more layers. It is designed on the premise that back-to-back layers can achieve a large effective receptive field, which reduces the total number of parameters needed.

3) GoogleNet: GoogleNet [29] was the winner of the 2014 ILSVRC, which introduced the ‘Inception Module’ as a novel structure. Instead of stacking convolution and pooling layers sequentially, the proposed structure has convolution and pooling layers in parallel, which requires many fewer parameters than AlexNet.

4) ResNet: The Residual Network [30] was the winner of the 2015 ILSVRC, which introduced a residual structure connecting every 3 layers. The Residual Network may have a thousand layers and can be more robust to overfitting.

5) Locally Connected Layers: Locally connected layers [31] allow different weights when the filter is applied at different positions to the input. It was selected with hopes of recognizing the surrounding structures and their locations. The locally connected neural network used here is similar to AlexNet, however the number of filters has been reduced.

B. Training setup

Using a “dungeon map” auto generator from [32], we generated 10,881 random 2D maps. A typical map can be found in the first column of Figure 2. These maps all have the same resolution of 640 by 480, consisting of randomly-sized rooms connected by corridors. Every map is comprised of one single connected component. A mean image of all the maps we used in this study is shown in Figure 4. The map data set was split into two parts, training and testing, with a ratio of 9:1 for the respective sets. A robot is initialized at a random location in each map and takes the optimal action using the method described in Section II-B. If the optimal action’s information gain surpasses a threshold (the threshold is slightly greater than zero), the robot performs the action and adds the sample into the training or testing data set, depending on which set the map
has been assigned to. The robot will switch to a new map if there is no information gain after an action has been executed. The total number of training samples collected was approximately 185,000, which amounts to approximately 20 samples per map. The number of testing samples collected was approximately 18,000. Every sample generated was used in the respective training and testing phases for all networks.

In representing the map-derived images, usually an occupied cell will have a grayscale value of zero associated with its respective pixel, however we have added an offset of 5 to the occupied value to avoid confusion in training the neural network. The grayscale pixel values used to represent free and unknown grid cells were 255 and 127 respectively.

The neural networks described above were trained using an Nvidia Titan X graphics card with caffe [33], while the labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU. Labeling was performed in MATLAB on a Intel i7 4790K CPU.

IV. COMPUTATIONAL RESULTS

A. Accuracy in Testing Phase over Individual Samples

We compared “top-1” and “top-5” accuracy between the different neural networks described in Section III-A. Top-1 accuracy represents the percentage of instances in which the neural network chose the correct label for its testing samples (i.e., it chose the action offering the maximum expected information gain per Eq. (3)) , and top-5 accuracy represents the percentage of instances in which, if the neural network is allowed to pick 5 actions, one of them is the top-1 action.

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>AlexNet</th>
<th>VGG</th>
<th>GoogleNet</th>
<th>ResNet</th>
<th>Local</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 1</td>
<td>68.381</td>
<td>11.339</td>
<td>69.195</td>
<td>11.334</td>
<td>65.676</td>
</tr>
<tr>
<td>Top 5</td>
<td>97.405</td>
<td>20.449</td>
<td>97.692</td>
<td>38.892</td>
<td>96.286</td>
</tr>
<tr>
<td>Size</td>
<td>large</td>
<td>large</td>
<td>smallest</td>
<td>large</td>
<td>small</td>
</tr>
</tbody>
</table>

As shown in Table I, GoogleNet has the highest accuracy in both the top-1 and top-5 cases, and also requires fewer parameters than other formulations. AlexNet came very close in accuracy compared to GoogleNet, however it requires more parameters. A locally connected neural network gave slightly less accuracy than AlexNet, but requires fewer parameters.

Both VGG and ResNet are not learning much in this application, though both of them have great success in image classification challenges. We suspect that it may be preferable to have a larger filter size (e.g., the first filter layer of AlexNet is 11, and for GoogleNet it is 7). A small filter size (e.g., most of the VGG and ResNet filters have a filter size of 3) may not capture the features well enough in our case.

B. Application to Full Robot Exploration Sequences

In order to evaluate the performance of a trained neural network over full robot exploration sequences, we generated several new dungeon maps for comparison between GoogleNet, the best-performing framework over our 18,000 testing samples, and an exhaustive information-theoretic approach that selects sensing actions according to Equation (3). The robot was initialized from a random location and repeatedly queried the neural network for five actions; all five actions were evaluated for MI and the robot selected the action offering maximum MI. The baseline approach in this scheme was a mutual information controller which exhaustively evaluates the expected information gain of all the possible actions at every decision-making step (up to 36 actions when none are obstructed by obstacles). Note that we consider only the “next best view”, without providing a multi-step rescue action when the robot runs into a dead-end, which could work in concert with the proposed neural network enabled information-based controller.

![Figure 5: A comparison of GoogleNet with an exhaustive information-theoretic approach over a series of exploration trials paired with different randomly-generated maps, in which a robot explores until it fails to collect new information. The data has been normalized with respect to the information gain obtained from the exhaustive approach.](image-url)
is allowed to proceed until the robot encounters an iteration in which it fails to acquire new information. All trials are shown in their entirety in the video attachment.

Despite the 97% top-5 accuracy observed from GoogleNet, it does not achieve the same level of performance as the exhaustive information-theoretic approach over full robot exploration sequences. We suspect a contributing factor is that all of the training samples were generated after taking the optimal action at a previous step, and the neural network may struggle to recover after one or more sub-optimal actions are taken by the robot in practice. If we consider that, for each decision-making step, the neural network will be able to select a top-5 action 97% of the time, then the possibility of doing this 20 steps in a row drops to only 54.38%.

Although the neural network does not outperform the exhaustive mutual information controller with respect to entropy reduction, the former is more computationally efficient than the latter. We ran the exhaustive information-theoretic approach on Intel i3 3.7Ghz and i5 3.5Ghz CPUs; the neural network approach was run on Nvidia GTX 960 and Titan X GPUs (each paired with an i5 3.5Ghz CPU). After the neural network selected its top-5 actions (via the GPUs), the MI associated with all 5 actions was evaluated (via CPU), and the best action was selected. The average computation time per decision-making step of each exploration process is summarized in Table II, where the neural net times include the time for decision-making step for each exploration process. We show plots of a few intermediate filters of GoogleNet and AlexNet as well as the output from those filters in Figure 6. The first convolution layer of GoogleNet has 64 filters of size 7 by 7, which are combined into a 56 by 56 matrix shown in Figure 6b. The output of the first convolution layer was also combined into one matrix shown in Figure 6c. The second convolution layer of GoogleNet has 192 filters of size 3 by 3, which is shown in Figure 6d, and its output is shown in Figure 6e. The output of the first GoogleNet convolution layer indicates some edges (between free and occupied, and explored and unexplored space) being extracted by the neural network in Figure 6c, and the output of the second convolution layer, shown in Figure 6e, indicates the extraction of edges, and larger chunks of free, occupied, and unknown space.

V. Conclusions

We have proposed a novel approach to predict a robot’s most informative exploratory action using a deep neural network, for the purpose of exploring a priori unknown environments with a range-sensing mobile robot. The proposed approach is more efficient in the testing phase than an information-theoretic optimization approach, and achieves competitive performance that has the potential to guide an autonomous mobile robot, if used in concert with other standard heuristics that can “rescue” a robot from dead ends. Assuming that the required offline training resources are available, the proposed approach has the potential to scale to higher-dimensional robots, and 3D maps, by virtue of the ease of testing. However, the specific means for training a neural network to explore higher-dimensional action spaces and workspaces, and to perhaps consider longer sequences of exploratory actions, remain an area of interest for future work, as does supplementing our training data with flawed scenarios that reflect the occasional selection of sub-optimal actions. This may allow us to further bridge the gap between the fast and reliable selection of high-quality individual sensing actions detailed in Section IV A, and the execution of full exploration sequences detailed in Section IV B.

ACKNOWLEDGMENTS

The authors thank Jinkun Wang from the Mechanical Engineering Department of Stevens Institute of Technology for help with training data preparation. This research has been supported in part by the National Science Foundation, grant number IIS-1551391.

REFERENCES


C. Network Visualization

We show plots of a few intermediate filters of GoogleNet and AlexNet as well as the output from those filters in Figure 6. The first convolution layer of GoogleNet has 64 filters of size 7 by 7, which are combined into a 56 by 56 matrix shown in Figure 6b. The output of the first convolution layer was also combined into one matrix shown in Figure 6c. The second convolution layer of GoogleNet has 192 filters of size 3 by 3, which is shown in Figure 6d, and its output is shown in Figure 6e. The output of the first GoogleNet convolution layer indicates some edges (between free and occupied, and explored and unexplored space) being extracted by the neural network in Figure 6c, and the output

<table>
<thead>
<tr>
<th>CPU/GPU</th>
<th>CPUs (baseline)</th>
<th>GPUs (Neural Net)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T 1.7Ghz</td>
<td>i5 3.5Ghz</td>
</tr>
<tr>
<td>Time per step (seconds)</td>
<td>0.47</td>
<td>0.47</td>
</tr>
</tbody>
</table>

[1] Video attachment: https://www.youtube.com/watch?v=0j788bUq

ACKNOWLEDGMENTS

The authors thank Jinkun Wang from the Mechanical Engineering Department of Stevens Institute of Technology for help with training data preparation. This research has been supported in part by the National Science Foundation, grant number IIS-1551391.

REFERENCES

Fig. 6: Visualization of the intermediate layers of GoogleNet and AlexNet.