Learning Safety-Aware Policy with Imitation Learning for Context-Adaptive Navigation

Bo Xiong[1], Fangshi Wang[1], Chao Yu[2], Fei Qiao[3,*], Yi Yang[3], Qi Wei[3] and Xin-Jun Liu[2]

[1]School of Software Engineering, Beijing Jiaotong University, Beijing, China
[2]Department of Mechanical Engineering, Tsinghua University, Beijing, China
[3]Department of Electronic Engineering, Tsinghua University, Beijing, China
[*]Corresponding author: qiaofei@tsinghua.edu.cn

Abstract

This paper presents an Imitation Learning (IL) based visual navigation system, which could guide the robots navigating from some start position to a goal location without any explicit map. We pay close attention to the safety issue due to partially-observability and data distribution mismatching—when the robot meets some incomplete or unfamiliar states, it probably performs an unsafe action, making it hard to work on lifelong robot navigation. In this paper, a sequence-to-sequence (Seq2seq) deep neural network is built to enhance the agent's context-awareness in partially-observable conditions and boost the model's adaptability to unseen scenarios. Additionally, we propose Uncertainty-Aware Imitation Learning (UAIL) by explicitly estimating model uncertainty and actively request experts for labeling samples according to the uncertainty with On-Policy IL. Simulations demonstrated that the combined method—Safety-Aware Imitation Learning (SAIL) in goal-driven visual navigation achieves 35.6% shorter expected moving steps and 22% fewer collisions compared with current counterparts. With the learned safer policy, SAIL had be successfully adapted to unseen environments with minimal navigation performance loss.

1 Introduction

Considering a task of navigating from a current location to find a specific goal. Classical geometry-based methods, such as Simultaneous Localization and Mapping (SLAM) [1], [2], [3], [4], [5] can be divided into two stages: one stage is building a 3D map using imagery feature matching and geometry constraints, the other stage is global or local path planning. SLAM-based approaches require carefully designed image features and are hard to work on texture-less environments. Recently, learning-based approaches have dominated robot learning including manipulation [21, 22], self-driving cars [10, 15, 16] and robot navigation [19, 20]. Compared with traditional methods, learning-based navigation could work in an end-to-end fashion without any explicit map-building. One framework for doing this is Reinforcement Learning (RL) [6, 7, 8, 9]. A reward function is usually given in RL, then agents learn a policy to maximize the cumulative reward by interacting with the environment. However, designing such an appropriate reward function in real-world scenarios is too difficult for humans. Moreover, the sparse reward of RL in goal-driven task could lead to poor convergence and computation efficiency.

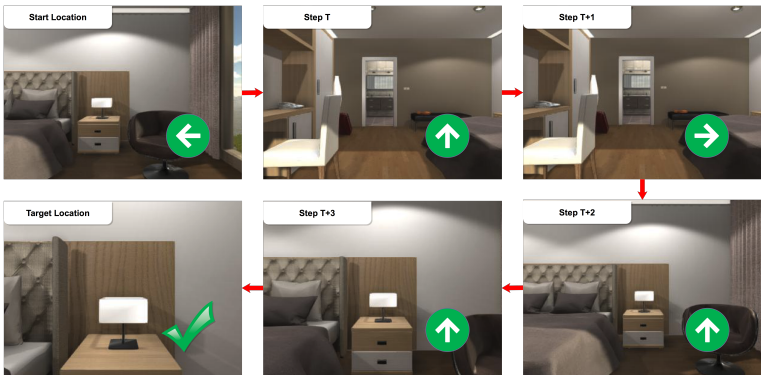

Figure 1: End-to-end target driven navigation task, the goal is to learn a mapping from observation to action which consumes minimum moving steps from a start position to the target location without any known map.

Alternatively, Imitation Learning (IL) [10, 11, 12] has been proposed to resolve reward-function-designing issues in RL. Rather than designing such a reward function, IL could learn policies directly from observing expert's demonstrations and generalize to new situations without any explicit interaction with environments. A common approach for IL is Behavior Cloning (BC) [13], where a robot observes a supervisor's policy and learns a mapping from states to actions directly with supervised learning. However, BC has a prerequisite that demonstrations must meet the i.i.d assumption of statistical learning, or will suffer from several problems—with the execution of the robot's policy, robot's state will move to a different distribution from teacher's demonstration which it was trained on, making it drift to dangerous states [14]. For instance, when a robot moving to a strange state, collisions could easily happen. Moreover, the robot's action estimation errors will compound once the robot's states drift away from the supervisor's demonstrations. Therefore, the model is hard to be adapted to context-changing environments, which prevents the application in life-long navigation. On-policy approach, such as Data Aggregation [14], can partially alleviate this issue by querying corrective samples online and iteratively aggregate new data for training. However, DAgger requires a huge number of queries to update its policy, which could be tedious for human teachers to answer and add more unnecessary computations. Recent works have paid more attention to safety issues in robot learning tasks, such as safe RL [43, 44, 45], but seldom consider the IL's safety in specific applications. In addition, the model's adaptability toward unseen environments plays a vital role in lifelong robot navigation, which should be explicitly modeled in the deep neural network of IL.

This paper presents Safety-Aware Imitation Learning (SAIL) framework by addressing both partially observability and data distribution mismatching in IL. Firstly, to enhance the context-awareness in partially observable environments and facilitate the adaptability to unseen scenarios and goals, we build a sequence-to-sequence deep neural network with Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). In this network, both spatial relevance and temporal relevance are taken into consideration, which could significantly enhance the agent's context-awareness and improve the model's generation performance toward unseen scenarios and goals. Secondly, UAIL is proposed using Bayesian approximation with MC-Dropout [42]. For those potentially uncertain or unsafe actions that occurred in some unseen or unfamiliar scenarios, rather than to perform it anyway, UAIL request for expert's advising (similar to active learning) for whether to act or label it. We predict the model uncertainty with MC-Dropout, a Bayesian approximation for uncertainty estimation in deep learning. This uncertainty has been used to improve the safety and efficiency of On-Policy Imitation Learning such as DAgger. Extensive experiments in the simulator have been conducted to compare the combined SAIL method with the vanilla IL approach, SAIL shows better navigation performance (35.6% fewer expected moving steps) and safer policy (22% lower collision rates). Additionally, with the learned safer policy, SAIL had shown successfully adapted to unseen scenarios and goals with minimal navigation performance loss. The main contributions of our work are listed as followings:

(1) We built a Seq2seq deep neural network to enhance the agent's context-awareness in partially-observable scenarios and facilitate the model's adaptability to unseen scenarios and goals.

(2) We presented Uncertainty-Aware Imitation Learning (UAIL) approach by estimating model uncertainty with MC-Dropout and combining it with On-policy IL method, which significantly improves the safety of IL.

(3) We proposed SAIL by combining UAIL and the Seq2seq network and done extensive experiments and evaluations, including navigation performance, safety, and the adaptability to unseen environments and goals.

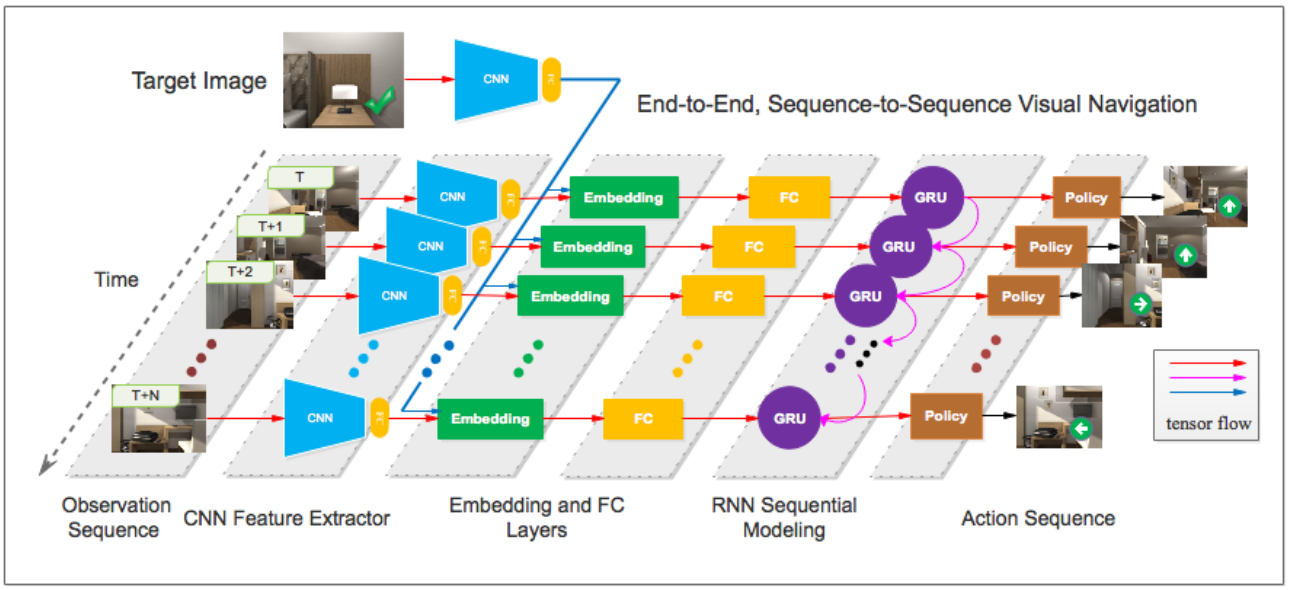

Figure 2: The proposed end-to-end, sequence-to-sequence neural network architecture for context-adaptive visual navigation. 1) we model the spatial relevance between observation and goal with Siamese CNN Network. in each time step, both of current observation and goal image are feed into the CNN network, where ResNet-50 are used to extract the visual features of images. 2) we model the temporal relevance between current observations and past observation it has seen before to address partially observability. GRU is used to maintain the hidden representations of the input observations it has seen before.

In the following sections, related work is presented in Section II. Section III demonstrates the SAIL framework. Experimental results are described in Section IV before making a conclusion in Section V.

2 Related Work

This work is relevant to past literature in the domains of imitation learning, learning-based navigation and uncertainty estimation. In this section, these areas' related work is reviewed respectively.

2.1 Imitation Learning

One of the commonly used solutions for IL is Behavioral Cloning (BC) [13], where the robot passively observes expert's full demonstrations and learns a policy mapping state to action via purely supervised learning. However, BC suffers from serious safety problems, when executing its policy, the robot will drift to dangerous states. For example, when a self-driving car steers to the edge of the road, it cannot be able to recover from it [14]. [23] has pointed out it was due to the robot's distribution being different from its demonstrator's, once robot drifting away from expert's demonstrations, robot's error will compound, which is a known problem named compounding error (or covariate shift). DART algorithm [24], where noise is injected into the expert's trajectory, can make the gap of distribution more nearly, but DART cannot work on the scenario where the gap is serious. Inverse Reinforcement Learning (IRL) [25] could restore the reward function from the expert's behavior trajectory, which enables RL in turn. [26], [27] are two different branches of IRL. Generative Adversarial Imitation Learning (GAIL) [28] build a generator and a discriminator to find a strategy that matches the distribution of state-action pairs of experts and does not require any assumptions about the environment. The common problem of IRL and GAIL is their poor scalability in real-world settings and expensive computation. On-policy approaches have been proposed to address the compounding error problem in off-policy IL. DAgger [14], [15] (Data Aggregation) is one of the classical on-policy solutions. By continuously querying the experts for new corrections during execution, it can make the robot's execution trajectory distribution closer to the supervisors'. which has been demonstrated to reduce compounding error and learn robust policies. However, DAgger suffers from several limitations: 1) it is difficult for human experts to provide enough labeling; 2) visiting highly sub-optimal states is potentially dangerous for a robot in real settings [29]; 3) it is computationally expensive to iteratively update the policy.

To alleviate the computation burden, [29] tries to train a classifier to predict whether or not is safe during the robot's execution. However, this approach adds challenges to computational efficiency and human experts.

2.2 Learning-based Navigation

Learning-based techniques, especially the deep learning, approach the visual navigation problem in an end-to-end fashion. There are mainly two types of learning based visual navigation methods: 1) RL based methods [6, 7, 8, 9]: RL based methods are divided into two stages: an exploration stages, where the environment's map information is implicitly gathered; and an exploitation step, where the map information is used to navigate efficiently [30], The paper [6] explored deep RL for target-driven navigation—navigating from a current location to a target position. The main objective of goal-driven navigation is to find the minimal sequence of actions from its current location to a target. However, this method requires 100 million frames to converge, which is too difficult to train; [30] applied an LSTM extension to DRL for 3D maze navigation. 2) IL-based methods: [32] proposed an imitation learning method for autonomous control of an aerial vehicle. [33] explored 3D navigation tasks such as 2D grid navigation, target-reaching and line-following with deep IL. [32] applied the IL to autonomous navigation in complex natural terrain. [34] proposed zero-shot IL for visual navigation. All of the IL-based methods suffer from either unsafe problem due to data distribution mismatch or poor computation efficiency.

2.3 Uncertainty Estimation in Deep Learning

There are two types of uncertainty in deep learning: 1) Aleatoric uncertainty, or statistical uncertainty, is resulted from the intrinsic data distribution themselves [46,47]. There are some approaches [48] had been used to model aleatoric uncertainty, but it cannot be alleviated by simply adding more samples or prior knowledge to the deep learning system. 2) Epistemic uncertainty, as known as systematic uncertainty, come from the model parameters themselves due to limited it theoretically could be eliminated by giving additional training samples or offering more prior knowledge. [50] provide a deep analysis of estimating uncertainty in deep neural networks. One of the most popularly used frameworks for estimating epistemic uncertainty in deep learning is Bayesian approximation. Such as MC-dropout [46, 49] and Bayesian ensemble [51]. In this work, we applied MC- Dropout to estimate the model uncertainty due to its plug-and- play advantages in existing deep network architectures. The only thing we need to do is to pass the same input to the deep network multiple times with random dropout rate and compute the entropy or variance of the outputs, which could be used to represent the uncertainty.

3 SAIL METHOD

In this section, the SAIL framework for context-adaptive navigation is described in detail. Firstly, we introduce context-aware deep neural network design for goal-driven navigation, then the safety aware imitation learning using uncertainty estimation with MC-Dropout is presented.

3.1 Context-Aware Neural Network Design

For network design, as shown in Fig.2, we consider the spatial relevance between the observations and goal image. In each time step, both of the current observation and goal image are fed into the network. The intuition of doing this is inspired by [6], [40], which can be used to improve the model's generalization performance on unseen goals. When applying the IL approach to unseen goals, the network needs no re-training for each goal. Furthermore, we also take into account the temporal relevance of state-action pairs before the current observation. This technique works especially on scenarios where the robot can hardly observe all of the details of the current state (or Partially-Observable Markovain Decision-Making). Recurrent Neural Network (RNN) is used to address this problem. The goal of the network is to estimate the current action (such as moving forward or turning right) from all of the history states and goal together. To be specific, this network can be divided into three parts.

CNN Feature Extractor: To extract the image features of observations and targets respectively. we use two separate weights-shared CNN streams—ResNet-50 [35]. which are used to transform the two images into the same embedding space. ResNet-50 is pre-trained on ImageNet [36] and finetuned on our dataset before the training stage. The outputs of the ResNet-50 are projected into a 512-dimension space.

Embedding and FC Layer: this layer is used to fuse the observation feature and goal image feature to a 1024- dimension joint representation and then projected it to a new 512-dimension vector on fully-connected fusion layers. We use two separate fully-connected layers to learn the joint feature representation of observation and goal image.

RNN Context Modeling: To learn the temporal relevance of observation sequences. RNN Layers is added right after the embedding and FC layer. Gradient Vanishing is a key problem of RNN with long-term dependencies. Although LSTMs [38] are more prevalent in addressing this problem in past literature, we make use of GRUs [37] that have smaller number of parameters, simpler to use and are generally faster to train than LSTMs.

A sequence-to-sequence IL model is built as Fig 2. At the time step $t = 0$, an initial goal g and a start state s_0 are fed into the network, then in each time step $t > 0$, the agent takes an action at with its current policy π_θ.

$$a_t = \pi_\theta(s_t, g) \tag{1}$$

Then the environment changes its state to a new state s_{t+1} according to the transition dynamic env.

$$s_{t+1} = env(s_t, a_t, g) \tag{2}$$

The current hidden vector h_t is a function f_θ on current observation st, current goal g and the previous hidden vector h_{t-1}, θ is the parameter of function f_θ.

$$h_t = \begin{cases} s_0 & t = 0 \\ f_\theta(h_{t_1}, s_t) & t >= 1 \end{cases} \tag{3}$$

We can predict an optimal action $a_t^\wedge$ with $P_\theta(a_t|h_t)$.

$$a_t^\wedge = arg\max_a P_\theta(a_t|h_t) \tag{4}$$

where

$$P_\theta(a_t|h_t) = softmax(h_t) \tag{5}$$

Given the training set $D = (s^i, a^i)$ The goal is to maximize the log-likelihood of the output action sequences.

$$\theta^* = arg\max_\theta log \sum_{(s^i,a^i)\in D} P_\theta(a^i|s^i) \tag{6}$$

where

$$\log P_\theta(a|s) = \sum_t \log P_\theta(a^t|s^t) \tag{7}$$

We minimize the cross-entropy loss between predicted action and corrective action made by experts with Adam optimization algorithm.

$$L(\theta) = \sum_{\tau in D} \sum_a [\pi_E(a|s, g) \log(\pi_\theta(a|s, g))] \tag{8}$$

3.2 Safety-Aware Imitation Learning

Considering the safety issues of off-policy IL due to the data distribution mismatching. UAIL was proposed by considering model uncertainty and combining it with vanilla on-policy training methods—Data Aggregation (DAgger) [14], [15]. UAIL iteratively query the expert but not frequently as DAgger for new correctives samples to train IL model. The uncertainty information in the current model is used to guide UAIL whether or not to ask the expert for new sample labeling. Therefore, this combined on-policy IL method is query-efficient, only when the uncertainty value is greater than a certain threshold, then we ask the expert for labeling and vice versa. when the accumulated new labeled data is enough for training a new policy (the number of new labeled data is larger than another threshold), we aggregate the new labeled data with the old one and retraining our network using the new aggregated dataset. A basic idea to estimate the model uncertainty in neural networks is using the entropy of softmax output in the categorical neural network. This method is very naïve because it always happened that the softmax entropy is large but the real uncertainty is small and vice versa.

Uncertainty Estimation with MC-Dropout: To bridge the gap between uncertainty estimation and deep neural network, we use MC-Dropout, a Bayesian approximation of uncertainty estimation in deep learning to represent the confidence of action to be executed. Basically, it has been demonstrated that using Dropout

at inference time in the deep neural network is equivalent to doing Bayesian approximation in Bayesian deep learning. The key idea here is letting dropout doing the same thing in both training and testing time. At test time, we will repeat *beta* times in passing the same input to the network with a random dropout value. MC-Dropout provides an efficient way to estimate uncertainty with minimal changes in most existing deep networks. It provides a plug-and-play module to deep learning for uncertainty estimation.

Safety-Aware Imitation Learning: To improve the training efficiency of the on-policy training approach, we intend to reduce the number of queries and aggregation times for data aggregation. We propose an uncertainty-aware imitation learning method (similar to active learning). It initializes model's weights on initial dataset with supervised learning, then executes the current policy until the learner's confidence falls below a solid threshold, at which point it queries the expert for corrective action. Uncertainty based approach may decide to stop asking for queries once the confidence exceeds the threshold in all states. which make use of the complementary advantages of humans and robots. Algorithm 1 summarizes the training procedure of Safe IL with uncertainty estimation.

Algorithm 1 SAIL: Safety Aware Imitation Learning.

Input: Initial demonstrations, D_0; uncertainty threshold, $\epsilon_{uncertainty}$; data aggregation threshold $\epsilon_{aggregation}$; aggregation episodes, M; batch size,N;
Output: policy parameter, θ;

$\theta_0 = SupervisedImitationLearning(D_0)$;
for $I = 0; M$ **do**
  for $t = 0; N$ **do**
    $a_{t+1,\theta} = PolicyExecute(\pi_{\theta_t}, s_{i,t}, g)$
    $\epsilon_{uncertainty} = MCDropout(a_t, s_{i,t}, g)$
    if $a_t < \epsilon_u ncertainty$ **then**
      $a_{t+1,E} = QueryExpert(s_{i,t}, g)$
      $D_t, T = AggregationPool(a_{t+1,E}, s_{i,t}, g)$
    else
      if $T > \epsilon_{aggregation}$ **then**
        $D_{t+1} = DataAggregation(D_0, D_t)$
        break
      else
        return (4)
      end if
    end if
  end for$SupervisedImitationLearning(D_{t+1})$
end for
$return\theta$

4 Experiments

In this section, we evaluate the SAIL approach in the simulation environment—AI2-THOR1 [39]. We start out by introducing the dataset and experiment setup, then present extensive experimental results on several metrics.

4.1 Dataset

AI2-THOR framework is used to collect the training data. AI2-THOR is an excellent photo-realistic interactive 3D simulator for AI agents, it provides a 3D environment that looks similar to the real-world scenes. There are totally 120 scenes in the AI2-THOR environment covering 4 different environment types, including kitchens, living rooms, bedrooms, and bathrooms, and each room type consists of 30 specific scenes. In AI2-THOR, the agent's positions are sampled from the discrete grids with 4 action spaces (move forward, move back, turn right and turn left). Each image consists of the agent first-person view in the rooms.

4.2 Experiment Setup

In this experiment, the authors use 4 different scenes to train our models (bathrooms, bedrooms, kitchens, and living rooms). For expert's policy generation, we use the A* search algorithm to find the shortest path from the start location to the target location while avoiding collisions simultaneously. In order to narrow the situation

gap between different room types, we trained our model on each scene separately. In each scene, the authors set 10 remarkable goal locations, such as the laptops, chairs and table lamps. For each goal, this work runs 50 stochastic starting points to evaluate the performance. To validate the proposed approach, the authors compare and evaluate the following 4 baselines and 2 proposed methods.

(1) BC (off-policy): an off-policy IL algorithm that learns a mapping from states to actions directly by supervised learning without any demonstrations sampling.

(2) DART (off-policy): an improved BC method that randomly injects noise into the expert's trajectory.

(3) DAgger (on-policy): an on-policy training approach by iteratively querying experts for the new correct sample and aggregating the new dataset for training.

(4) SaferDAgger (on-policy): an extension of vanilla DAgger method that minimizes the number of queries to a reference policy both during training and testing stage.

4.3 Evaluations and Results

The ultimate goal of goal-driven navigation is to find a given target with minimum steps while avoiding collisions simultaneously. Although these two facts can be affected by each other, having fewer moving steps is not equal to having fewer collisions in real scenarios. Therefore, we evaluate SAIL's performance from several different metrics.

Expected Trajectory Steps. We evaluate the navigation performance by expected trajectory steps, the expected trajectory steps are defined as the total number of steps taken to navigate from an initial start position to a given target location. The goal of this task is to minimize the expected trajectory steps, the closer the expected navigation steps to the shortest path is, the better the navigation's performance is. For four different scenes, the authors compare the training results of 4 different scenes with shortest path steps, as shown in Table 1. It can be seen that SAIL with MC-Dropout achieve shorter expected moving steps than vanilla off-policy and on-policy IL approaches.

Navigation Safety Evaluation. In order to evaluate the robot's safety awareness in goal-directed navigation, the expected collision rate is defined as the mean percentages of robot's collisions with obstacles when navigating from different start locations to different target positions. In this work, the agent is allowed to collide with the environment, when the agent collides with the environment, we recognize it as one navigation step. As shown in Table 2. SAIL with MC-Dropout can significantly reduce the collision rate of goal-driven navigation.

Navigation Success Rate in Unseen Scenes. Another metric to evaluate navigation performance is the navigation success rate in unseen scenes. Since there are 120 different scenes in each types of environments, we set 100 scenes as training scenes and 20 scenes as testing scenes. Experiment result (see Table 3) show that the performance of SAIL in unseen scenes was farther markedly improved than the baseline IL model. This can be explained by the safer policy learned by SAIL and improved perception ability due to context-aware neural network design.

Table 1: Expected moving steps and collision rate for goal-driven navigation.

Method	Bathroom	Kitchen	Bedroom	Living room
Shortest Path	7.11	10.87	15.37	16.02
BC [13]	15.58	37.19	55.95	66.34
DART [24]	15.51	36.62	53.38	63.21
DAgger [14]	14.31	35.52	51.39	55.21
SaferDagger [29]	13.34	25.01	46.46	48.31
SAIL(Entropy)	13.34	25.01	46.46	48.31
SAIL (MC-Dropout)	12.21	17.35	33.55	37.32

5 Conclusion

This work proposes the SAIL approach for goal-driven context-adaptive visual navigation. This approach addresses the robot's safety issues due to data distribution mismatch and poor performance on partially-observable visual navigation. To alleviate the safety issue, we model the policy uncertainty in deep learning with MC-Dropout and combine it with the on-policy IL approach. To enhance the agent's perception capability and context awareness in partially-observable environments, we build a sequence-to-sequence deep neural network

Table 2: Expected collision rate for goal-driven navigation.

Method	Bathroom	Kitchen	Bedroom	Living room
BC [13]	0.30	0.43	0.49	0.52
DART [24]	0.28	0.41	0.42	0.45
DAgger [14]	0.29	0.42	0.45	0.47
SaferDAgger [29]	0.26	0.38	0.41	0.44
SAIL(Entropy)	0.26	0.32	0.39	0.41
SAIL (MC-Dropout)	0.24	0.31	0.35	0.37

Table 3: Navigation success rate in unseen scenes,where the maximum penalty factor is 5 times the steps of shortest path.

Method	Bathroom	Kitchen	Bedroom	Living room
BC [13]	0.89	0.83	0.71	0.64
DART [24]	0.91	0.87	0.74	0.67
DAgger [14]	0.88	0.86	0.74	0.68
SaferDagger [29]	0.96	0.91	0.81	0.69
SAIL(Entropy)	0.95	0.94	0.89	0.75
SAIL (MC-Dropout)	0.96	0.98	0.92	0.81

with both spatial and temporal relevance taken into consideration. Experiments on simulator demonstrated that the proposed SAIL method had better navigation performance on several evaluation metrics, compared with the state-of-the-art IL methods. With the safer policy, SAIL had shown successfully be adapted to unseen scenarios and goals with fewer collisions and minimal navigation performance loss.

Acknowledgement

We are thankful to Ai Shi who moderated this paper and, in that line improved the manuscript significantly.

References

[1] Mur-Artal R, Tardós J D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras[J]. IEEE Transactions on Robotics, 2017, 33(5): 1255-1262.

[2] Davison A J, Reid I D, Molton N D, et al. MonoSLAM: Real-time single camera SLAM[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2007 (6): 1052-1067.

[3] Engel J, Schöps T, Cremers D. LSD-SLAM: Large-scale direct monocular SLAM[C]//European Conference on Computer Vision. Springer, Cham, 2014: 834-849.

[4] Endres F, Hess J, Sturm J, et al. 3-D mapping with an RGB-D camera[J]. IEEE Transactions on Robotics, 2014, 30(1): 177-187.

[5] Yu, Chao, Liu, Zuxin, Liu, Xinjun, et al. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments[J]. 2018.

[6] Zhu Y, Mottaghi R, Kolve E, et al. Target-driven visual navigation in indoor scenes using deep reinforcement learning[C]//Robotics and Automation (ICRA), 2017 IEEE International Conference on. IEEE, 2017: 3357-3364.

[7] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In International Conference on Machine Learning, 2016

[8] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard. Deep reinforcement learning with successor features for navigation across similar environments. arXiv preprint arXiv:1612.05533, 2016.

[9] J. Zhang, L. Tai, Y. Xiong, M. Liu, J. Boedecker, and W. Burgard. Vr goggles for robots: Real-to-sim domain adaptation for visual control. arXiv preprint arXiv:1802.00265, 2018.

[10] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. arXiv preprint:1604.07316, 2016.

[11] Giusti A, Guzzi J, Ciresan D C, et al. A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots[J]. IEEE Robotics and Automation Letters, 2016, 1(2): 661-667.

[12] N. D. Ratliff, J. A. Bagnell, and S. S. Srinivasa. Imitation learning for locomotion and manipulation. In International Conference on Humanoid Robots, 2007.

[13] Michael Bain and Claude Sommut. A framework for behavioural claning. Machine Intelligence 15, 15:103, 1999.

[14] S. Ross, G. J. Gordon, and J. A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. arXiv preprint arXiv:1011.0686, 2010.

[15] Codevilla F, Müller M, Dosovitskiy A, et al. End-to-end driving via conditional imitation learning[J]. arXiv preprint arXiv:1710.02410, 2017.

[16] Pan Y, Cheng C A, Saigol K, et al. Agile Off-Road Autonomous Driving Using End-to-End Deep Imitation Learning[J]. arXiv preprint arXiv:1709.07174, 2017.

[17] Giusti A, Guzzi J, Ciresan D C, et al. A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots[J]. IEEE Robotics and Automation Letters, 2016, 1(2): 661-667.

[18] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert. Learning monocular reactive UAV control in cluttered natural environments. In ICRA, 2013.

[19] Pathak D, Mahmoudieh P, Luo G, et al. Zero-shot visual imitation[C]//International Conference on Learning Representations. 2018.

[20] Tai L, Zhang J, Liu M, et al. Socially-compliant navigation through raw depth inputs with generative adversarial imitation learning[J]. arXiv preprint arXiv:1710.02543, 2017.

[21] N. D. Ratliff, J. A. Bagnell, and S. S. Srinivasa. Imitation learning for locomotion and manipulation. In International Conference on Humanoid Robots, 2007.

[22] P. Englert, A. Paraschos, J. Peters, and M. P. Deisenroth. Model-based imitation learning by probabilistic trajectory matching. In ICRA, 2013.

[23] S. Ross and D. Bagnell. Efficient reductions for imitation learning. In International Conference on Artificial Intelligence and Statistics, pages 661–668, 2010.

[24] Laskey M, Lee J, Fox R, et al. Dart: Noise injection for robust imitation learning[J]. arXiv preprint arXiv:1703.09327, 2017.

[25] Abbeel P, Ng A Y. Inverse reinforcement learning[M] //Encyclopedia of machine learning. Springer, Boston, MA, 2011: 554-558.

[26] Ziebart B D, Maas A L, Bagnell J A, et al. Maximum Entropy Inverse Reinforcement Learning[C]//AAAI. 2008, 8: 1433-1438.

[27] Ramachandran D, Amir E. Bayesian inverse reinforcement learning[J]. Urbana, 2007, 51(61801): 1-4.

[28] Ho J, Ermon S. Generative adversarial imitation learning[C]//Advances in Neural Information Processing Systems. 2016: 4565-4573.

[29] J. Zhang and K. Cho. Query-efficient imitation learning for end-to-end autonomous driving. arXiv preprint arXiv:1605.06450, 2016.

[30] Dhiman V, Banerjee S, Griffin B, et al. A Critical Investigation of Deep Reinforcement Learning for Navigation[J]. arXiv preprint arXiv:1802.02274, 2018.

[31] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J. Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, Dharshan Kumaran, and Raia Hadsell. Learning to navigate in complex environments. 2017.

[32] Sammut C, Hurst S, Kedzier D, Michie D et al (1992) Learning to fly. In: Proceedings of the ninth international workshop on machine learning, pp 385–393

[33] Hussein A, Elyan E, Gaber M M, et al. Deep imitation learning for 3D navigation tasks[J]. Neural computing and applications, 2018, 29(7): 389-404.

[34] Silver D, Bagnell J A, Stentz A. Applied imitation learning for autonomous navigation in complex natural terrain[C]//Field and Service Robotics. Springer, Berlin, Heidelberg, 2010: 249-259.

[35] Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, inception-resnet and the impact of residual connections on learning[C]//AAAI. 2017, 4: 12.

[36] Olga Russakovsky, Jia Deng, Hao Su, et al. ImageNet Large Scale Visual Recognition Challenge[J]. International Journal of Computer Vision, 2015, 115(3):211-252.

[37] Jozefowicz R, Zaremba W, Sutskever I. An empirical exploration of recurrent network architectures[C]// International Conference on International Conference on Machine Learning. JMLR.org, 2015:2342-2350.

[38] Sak H, Senior A, Beaufays F. Long short-term memory recurrent neural network architectures for large scale acoustic modeling[C]//Fifteenth annual conference of the international speech communication association. 2014.

[39] Kolve E, Mottaghi R, Gordon D, et al. AI2-THOR: An interactive 3d environment for visual AI[J]. arXiv preprint arXiv:1712.05474, 2017.

[40] Sadeghi, F., Toshev, A., Jang, E., & Levine, S. (2018). Sim2Real Viewpoint Invariant Visual Servoing by Recurrent Control. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4691-4699).