Summary of Energy-Efficient 3D Vehicular Crowdsourcing For Disaster Response by Distributed Deep Reinforcement Learning

John Dewey, jcd18c
jcd18c@fsu.edu

1 INTRODUCTION

Natural disasters, such as earthquakes, hurricanes, and explosions, cause large amounts of damage, injuries, and casualties in short periods of time. Urgent situations require immediate action by rescue teams, as it is pivotal to do as much as possible in the "golden 72-hours" since after which survival rate falls to about 5-10%. Use of unmanned vehicles (UVs) can improve efforts in disaster response as their deployment flexibility and information collecting ability can reduce amount of work required from people, explore disaster areas that are impossible for humans to explore, and focus the efforts of humans on rescue. However, the amount of research in dynamic UB trajectory planning in extreme conditions is sparsely researched, and there are a few challenges that must be addressed to using UVs to explore disaster zones such as making UVs explore the complete disaster area, making UVs cooperate, and planning trajectories of UVs in disaster areas with unevenly distributed point-of-interests (PoIs). To address this, DRL-DisasterVC(3D) is proposed.

2 RELATED WORK

2.1 Spatial Crowdsourcing (SC)

SC has been widely studied in both theoretical and various inductrial applications like web mapping services, ride-hailing services, and online search and recommendation systems. One of the key issues in SC is task assignment (TA) where workers are allocated to spatial tasks to maximize/minimize a total weighted value. Two categories of TA are online and offline scenarios. For online scenarios, Liu et al. proposed a threshold-based greedy algorithm in "Budget-aware online task assignment in spatial crowdsourcing." For offline scenarios, Li et al. proposed a 3D stable spatial matching solution in "," and Ni et al. proposed a game-theoretic approach to find an optimal worker-task routing path that considers task dependencies in "".

2.2 Deep Reinforcement Learning (DRL)

Reinforcement learning (RL) is widely used for sequential decision-making problems through iteratively interacting with a time-slotted environment, and are usually formulated by a Markov Decision Process (MDP). Actions are generated by a policy where a state transitions to its next state with a reward. DRL bridges RL and deep neural networks (DNNs) since DNNs allow the ability to learn intricate patterns and representations. Some representative DRL approaches include DQN, Rainbow, A3C, and DPPO; however, the state of the art approach is IMPALA - the core of DeepMind AlphaStar. IMPALA simultaneously increases speed and decreases instability of DRL training, and it is considered the starting point of DRL-DisasterVC(3D).

3 PROBLEM FORMULATION

3.1 System Model

The vehicular crowdsourcing (VC) task considers a set of UVs (drones and unmanned ground vehicles), $\mathcal{U}$, and a set of PoIs, $\mathcal{P}$. The UVs work to explore the PoIs in a 3D workzone that contains a set of obstacles to avoid while exploring. The task duration is fixed and divided into T equal timesteps of τ length, and each timestep is contains two parts, UV movement and data collection. The UV movement phase, a UV moves from its current position to a new position using an agle vector comprised of a polar and azimuthal angle and a moving distance (movement velocity is fixed). In the data collection part, a round robin sensing policy is used to collect data from a number of the nearest PoIs to a UV. Under the assumption that PoIs have multiple antennas using orthogonal frequencies, transmissions from PoIs will not interfere with eachother, so only the large scale pathloss effect between a PoI and UV is considered when measuring signal to noise ratio (SNR) and transmission rate. If the SNR is below a threshold, the data is considered too noisy and unusable. Therefore, the amount of data collected relies on data collection time, transmission rate, data dropout amount, and the number of PoIs serviced.

3.2 Problem Definition

Four metrics are used to define the UV navigation problem: data collection ratio, data dropout ratio, geographical fairness, and energy efficiency. The data collection ratio measures the average ratio of data collected from all UVs and the initial amount of data available data at all PoIs apon task completion, the data dropout rate measures data loss and the quality of the data collection due to impact of low SNR, geographical fairness uses Jain's fairness index to measure diversity and uniformity in the disaster workzone, and energy efficiency measures energy consumption during the task and combines the previous three metrics to achieve the goals of each simultaneously. This allows the energy efficiency metric to measure the overall performance of the task; therefore, the optimization problem can be viewed as maximizing energy efficiency. However, maximizing

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

4 SOLUTION

DRL-DisasterVC(3D) is a heuristic DRL method that consists of a distributed, asynchronous DRL framework. The framefork is based on IMPALA and uses a repetetive experience relay for improved learning efficiency an attentive 3D convolutional neural network (CNN) with auxiliary pixel control for spatial exploration.

4.1 Distributed DRL Framework with Repetetive Experience Relay (RER)

A multi-actor-one-learner architecture is adapted from IMPALA where multiple actors (UVs) asynchronously work to generate MDP tuples by interacting with an environment. However, in IMPALA each MDP tuple is only used once, and the quality of learning from sampled experiences determines the accuracy and speed of training. Time and accuracy are essential in the domain of disaster recovery, so the team behind the paper introduces RER to enhance learning stability. There RER stores b_M batches of experiences and can traverse these experiences b_k times. After visiting a batch, the RER scrolls to a new batch. Everytime a batch is visited, a counter is updated, and when the counter reaches b_k, the batch is thrown away and replaced with a new one. One thing to note is that the dimmensions of the action space for multiple UVs expands tremendously compared to a single UV case, and this expansion leads to larger differences in local policies used by UVs. To stabilize this, a clipped target network is introduced. The target network of policies is periodically synchronized with the learner's policy network output and policy update speed is limited using truncated importance sampling and a clipped ratio. When the clipped ratio is high, learning is fast but unstable, and as the clipped ratio is lowered, stability increases and speed decreases.

4.2 Attentive 3D CNN convolution with Auxiliary Pixel Control (PC)

In DRL-DisasterVC(3D), two 3D CNN layers are used to calculate spatial feature representations of an input state. Since relationshops exist between different spatial dimmensions, a multi-head-relational attention (MHRA) module is placed after each CNN layer in the model to better extract them and make more reasonable trajectories for UVs. In the MHRA module, a number of independent heads with their own convolution filters are used to produce queries, keys and values that indicate relational semantics. The output of all the heads are concatenated and passed to the next layer in the model. Layer normalization and a shortcut to connect the input/output of the attention module are used to speed up model convergence. Furthermore, since PoI distribution is uneven and obstacle position is random, PC is introduced to increase geographical fairness.
FINISH PC SECTION

4.3 Algorithm Description

There are two parts to the algorithm, multiple actors and a central learner. The actors have a shared RER and asynchronously generate experiences for the central learner. The actors take in a local environment and policy network. Each actor has a buffer that stores multiple n-step trajectories collected from the local environment, and after the worker interacts with the environment n times, it sends the experience in a batch and starts the next timeslot. When the buffer is full, it is sent to the RER, and if a broadcast hyperparameter is received from the central learner, the actor synchronizes its local policy with it. The central learner takes in a policy network, value network, PC network, and the target networks of the three previous networks. It starts by randomly initializing network weights and initializing each of the target networks, then it initializes the RER. During each gradient descent step, the learner samples a batch from the RER and computes a policy loss, value loss, entropy loss, and pixel loss that are used to make a total loss. The learner then uses gradient descent to optimize the total loss, updates the counter for the batch in the RER it visited, and discards and replaces a batch if its counter reaches the max number of visits allowed. After certain time thresholds are met, the central learner updates its target networks and broadcasts network weights to actors.

5 EVALUATION

5.1 DisasterSim

To evaluate the proposed model, the team behind the paper designed a simulator (DisasterSim) for VC in disaster response scenarios. DisasterSim was built using Unity 2018.3.14f1, Python 3.7.7m and Tensorboard 2.3.0. There are three layers to the simulation, the data layer, model layer, and visualization layer. The data layer contains scene configuration such as scene assets, system parameters, and task settings, and model data (DNN structures, optimizers, and network parameters). The model layer contains four modules, the scene generator, hyperparameter tuning, model training, and testing modules. The scene generator loads scene configurations from the data layer and creates the 3D scene,the hyperparameter tuning module finds suitable hyperparameters by changing scene configuration and re-running model training, the model training module handles training the model, and the testing module handles model testing. The visualization layer provides a visual of the simulation in Unity.

5.2 Experiment

The team conducted an ablation study to measure the effectiveness of MHRA and PC and tested how hyperparameter tuning can affect performance. The study considered DRL-DisasterVC(3D) with all components, without PC, without MHRA, and without both. Furthermore, different hyperparameter settings, specifically the number of heads used in MHRA and the number of traversals allowed for a batch in the RER, were tested to find an optimal settings to improve performance. In hyperparameter tuning test, other hyperparameters not mentioned follow the common settings previously used in IMPALA. Finally, DRL-DisasterVC(3D) was tested against five baselines, IMPALA, IMPACT, CA2C, Shortest Path, and Random.

5.3 Results

Overall results were very promising. In the ablation study, it was found that the data collection ratio and geographical fairness of the model increased when using PC showing that PC improves spatial exploration. Furthermore, energy efficiency significantly decreases when MHRA is not used when compared to when PC is not used.

This makes sense, as PC sacrifices energy efficiency to improve spatial exploration, and MHRA helps mitigate that shortcoming by extracting more spatial relational features. Also, When comparing DRL-DisasterVC(3D) with all components and without MHRA and PC, the version with all components performs 17.3% better confirming the benifits of using MHRA and PC together. In the hyperparameter tuning testing, it was found that using too many or too few heads with MHRA negatively affects performance. Using too few heads doesn't allow the model to properly extract more multi-level relational representations that help the model make better decisions, but using too many heads results in difficulties of model convergence due to a significant increased in number of network parameters. The same holds for the number of traversals for a batch in the RER, as too few traversals doesn't allow RER to improve sample efficiency and learning quality from limited experience, but using too many traversals overfits due to a shortage of experience diversity. Finally, when comparing against the baselines, DRL-DisasterVC(3D) outperforms each in terms of the dour metrics presented in the paper (data collection ratio, data dropout ratio, geographical fairness, and energy efficiency). Testing the impact of the number of PoIs shows that increased numbers can lead to increased data density in local areas which traps UVs into a local optimal without effective spatial exploration and relational representations created by the PC and MHRA respectively. Furthermore, testing the impact of the number of UVs shows that as number of UVs increases, so does data collection ratio and geographical fairness because more UV's naturally cover more area. However, energy efficiency is decreased due to multiplied energy consumption, but the MHRA and PC methods enforce UV cooperation which helps mitigate the drop in energy efficiency. Also to note, testing the impact of the SNR threshold shows that a high SNR threshold decreases data collection ratio and increases data dropout ratio monotonically, yet DRL-DisasterVC(3D) consistantly outperformed all other baselines regardless of SNR threshold. Finally, a complexity analysis shows that DRL-DisasterVC(3D) converges faster than the other DRL methods tested due to improved sample efficiency brought by the RER and learning stability brought by MHRA and PC, and the improved network structure causes no extra overhead as the running time of DRL-DisasterVC(3D) is almost identical to IMPACT and only slightly higher than IMPALA and CA2C.

6 STRENGTHS

The solution provided in the paper effectively solves or mitigates issues presented by the challenges introduced in the paper. Using MHRA and PC effectively ensures that UVs cooperate amongst eachother, fully explore a disaster area, and properly explore disaster areas with uneven distributions of PoIs. Furthermore, the additions to IMPALA done in the paper add little to no overhead, so the use of DRL-DisasterVC(3D) holds no time disadvantage compared to the current state of the art method. Besides this, DRL-DisasterVC(3D) outperforms all other baselines (including the state of the art method), so the model is seemingly a total improvement over current methods with no negatives.

7 WEAKNESSES

One aspect of the paper I found to be a weakness is the number of figures and level of explaination in the paper. While there are excellent figures in the paper that show how the model works at a high level or how the testing simulator works, some things in the paper just rely on equations and conceptual understanding to convey how a module/method works. For example, the paper gives a detailed explanation of the clipped target network and equations that show its target function, loss functions, etc.; however, reading it was initially not clear to me and I needed to review it several times before I felt I understood it. Furthermore, there are some assumptions taken in the paper that will not be applicable in a real-world scenario. For instance, it is assumed that UV movement velocity is fixed and PoIs are equipped with multiple antennas operating at orthogonal frequencies. Though these assumptions make sense for the proof of concept, if the work is to be brought to a real world setting, these assumptions must be stricken and handled.

8 YOUR SOLUTION

To improve the paper, I would begin by adding figures that show how various aspects of DRL-DisasterVC(3D) work more clearly such as the clipped target network, MHRA, and auxiliary PC. Next, I would analyze all the assumptions made in the paper, determine which are viable to explore for future research and improvement, and add a section to the paper outlining future/potential work. Finally, I would try to make a real world scenario for testing to add onto the simulation used for testing. For example, something like deploying a set of drones to explore marked PoIs in a warehouse that has obstacles strewn about.