IIS-AI Course: Final Project Guidelines

Proposal due: the end of Week 12
Progress report due: the end of Week 14
Poster session due: TBD, Week 16
Final project: Jan 20, 2019

November 28, 2018

1 Introduction

In the final project, you will work in groups of one, two, or three to apply the techniques that you've learned in this class to a new setting that you're interested in or develop new AI techniques for problems you are interested in. Which task you choose is completely open-ended, but you may get some inspiration from our example ideas.

Note that it will take several iterations to find the right project, so be patient; this exploration is an essential part of research, so learn from it. Have fun and don't wait until the last minute!

The final project should address of the following questions:

- What's the problem you aim to address?
- Why is this problem interesting and important?
- What are previous approaches for this problem and what are their limitations?
- What are the main contributions of your work?
- Is the contribution significant? why?
- How does the proposed approach address the limitations of previous approaches?
- What's the intuition behind the proposed approach and why it works?
- Is the proposed approach technical sound?
- How is the proposed approach is evaluated?
- What are the assumptions and limitations of this proposed approach?

2 Milestones

Throughout the quarter, there will be several milestones so that you can get adequate feedback on the project.

Proposal (10% Points) (2 pages max): Define your problems, survey existing works, discuss limitations of existing work, and describe the general idea of your approach.

Progress report (20% Points) (4 pages): Propose a model and an algorithm for tackling your problems. You should describe the model and algorithm in detail and use a concrete example to demonstrate how the model and algorithm work. You should also have finished implementing a preliminary version of your algorithm (maybe it's not fully optimized yet and it doesn't have all the features you want). Report your initial experimental results.

Poster session (20% Points): By the poster session, for empirical work, you should have finished implementation, run a good chunk of your experiments, and done some basic error analysis; for theoretical work, you should have finished main parts of the formal analysis. In the poster, you should describe the motivation, problem definition, challenges, approaches, results, and analysis. The goal of the poster is to convey the important high-level ideas and give intuition rather than be a super-detailed specification of everything you did (but you should still be precise). You will be evaluated on both the contents of the poster as well as your presentation. You should stand by your poster during your assigned slot. During the poster session, the course staff will come around. You should be able to give a 30-second elevator pitch providing the highlights of your work, but also be prepared to take questions about any of the specific details. Additional tips:

- Use lots of diagrams and concrete examples. Use bullets for the key points, and make sure you use a large font that can be read from a distance. Don't write long sentences and lots of complex equations.
- Organize your poster into sections, so it's clear what the components of the project are. This also makes it easy to delve into a particular component.
- Practice and polish your 30-second pitch. Someone should be able to understand what you're doing and importantly, why, from listening to it.
- If you can make a live demo, you should do it!
- At the beginning of the poster session, you should check in with your mentor, who will provide easels and stands in exchange for a group member's student ID card. You should setup your poster around the mentor's designated area.
- All group members are expected to be at the poster session. If you cannot make it, coordinate with your mentor.

- In all cases, submit your poster as a PDF by 11pm that evening.

Poster session peer review : During the time that you're not assigned, you should wander around and look at other people's posters ? after all, the whole point of having a poster session so that everyone can share what they've accomplished over the quarter. You might even get ideas for your own project. Based on your wanderings, you should choose 3 posters that you liked the most and write a sentence or two describing each poster and why you liked it. These reviews will not influence anyone's grade, but is just a fun way to encourage you to engage in the poster session.

Final report (50% Points) (5-10 pages): You should have completed everything.

3 Grading rubric

Task definition: is the task precisely defined and does the formulation make sense?

Approach: was a baseline, an oracle, and an advanced method described clearly, well justified, and tested?

Data and experiments: have you explained the data clearly, performed systematic experiments, and reported concrete results?

Analysis: did you interpret the results and try to explain why things worked (or didn't work) the way they did? Do you show concrete examples?

Extra credit: does the project present interesting and novel ideas (i.e., would this be publishable at a good conference)?

Regardless of the group size, all groups must do the same basic amount of work (e.g., oracles, baselines, error analysis) as described in each milestone. Of course, the experiments may not always be successful, so we will cut the smaller groups more slack, while larger groups are expected to be more thorough in their experiments.

Project Ideas

IIIS AI 2018

November 28, 2018

1 Object Segmentation for Atari Video Games

Design a self-supervsied approach for identify dynamic objects in a simple domain, i.e., Atari video games. You may gain some ideas from supervised learning approaches, e.g., Mask R-CNN.

2 Sample Efficiency of Reinforcement Learning

The state-of-the-art results of sample complexity for model-based reinforcement learning with discounted rewards are following: MoRmax (Szita & Szepesvari, 2010)[9] with compelxity $O(\frac{SA}{(1-\lambda)^6\epsilon^2)}log\delta^{-1})$ and UCRL-γ (Lattimore & Hutter, 2012)[5] with $O(\frac{N}{(1-\lambda)^3\epsilon^2)}log\delta^{-1})$ but assuming each state transitions to at most two states, i.e., $|supp(p(\cdot|s,a))| \leq 2, \forall(s,a)$ where N is the number of non-zero transitions. Can you design an model-based RL algorithm that have better sample complexity than MoRmax but with a relaxed assumption than UCRL-γ?

3 Multi-Agent Learning

Theoretical results of multi-agent learning are important yet challenging. In Lecture 9, we discussed that the basic gradient ascent algorithm does not always converge and the gradient ascent with policy prediction only converge in two-play, two-action, general-sum games. You can pick one of the following research topics:

Topic 1: the convergence of these two algorithms assume the learning rate are infinitesimal and each agent can observe the payoff matrix and the other agent's policy. Can you design a new algorithm to relax these assumptions?

Topic 2: Can you design a novel algorithm converging in 2x3 games (one agent has two actions and the other agent has three actions)? Is it also a no-regret algorithm?

The reference papers are [7], [12], [13], and [2].

4 Model-Free Reinforcement Learning with a Maximin Objective

In our lecture, we aims to find a policy for a Markov Decision Process (MDP) to maximize the expected discounted reward. Now we assume that there are multiple agents, where each agent has its own reward function. We use a multi-agent Markov decision process (MMDP) to this problem, defined by a tuple $\langle I, S, A, T, \{R_i\}_{i \in I} \rangle$, where

I $= \{1, \ldots, n\}$ is a set of agent indices.

S is a finite set of states.

A $= \times_{i \in I} A_i$ is a finite set of joint actions, where A_i is a finite set of actions available for agent i.

T: $S \times A \times S \rightarrow [0, 1]$ is the transition function. $T(s'|s, a)$ is the probability of transiting to the next state s' after a joint action $a \in A$ is taken by agents in state s.

R_i: $S \times A \rightarrow \Re$ is a reward function of agent i and provides agent i with an individual reward $R_i(s, a)$ after a joint action a taken in state s.

The problem is how to learn a joint decision policy π^* in a model-free way that maximizes the following maximin objective value function when the transition function T is unknown:

$$V(\pi) = \min_{i \in I} \mathbf{E}[\sum_{t=0}^{\infty} \lambda^t R_i(\mathbf{x}^t, \mathbf{a}^t)|\pi, b]. \tag{1}$$

where λ is the discount factor, the expectation operator $\mathbf{E}(\cdot)$ averages over stochastic action selection and state transition, b is the initial state distribution, and $\mathbf{x}^t$ and $\mathbf{a}^t$ are the state and the joint action taken at time t, respectively.

5 Learning to Plan

Learning to plan is an exciting area. There are several interesting works in this area, including Value Iteration Networks (2016 NIPS best paper)[10] and Universal Planning Networks[8]. Could you design a novel algorithm that can effectively learn to plan in challenging domains, like Atar games?

6 Hierarchical Reinforcement Learning for Sparse Rewards

Sparse reward is a fundamental challenging problem for RL. Hierarchical exploration approaches learns to select subgoals and how to achieve subgoals, which seems helpful for reinforcement learning with sparse rewards, as shown by Hierarchical Deep Reinforcement

Learning (Kulkarni et. al., 2016)[4], FeUdal Networks (Vezhnevets et. al, 2017)[11] and Data-Efficient Hierarchical Reinforcement Learning (Nachum et. al., 2018)[6]. These papers shows hierarchical exploration has some advantages over basic ϵ-greedy exploration, because it allows more quickly to explore regions far away from the initial state. Can you formally approve hierarchical exploration is more efficient than ϵ-greedy exploration for some settings, e.g., where states are not in a small world (i.e., the average path length between states are small)?

7 Hierarchical Reinforcement Learning for Long-Horizon Tasks or Lifelong Learning

As humans, we often learn basic skills when solving tasks, reuse these skills for other tasks later, and learn new skills when existing skills only solve partial of the new task. Some works in hierarchical reinforcement learning aims to address similar problems, including the option-critic architecture (2016 AAAI best paper) (Bacon et. al., 2016)[1] and Stochastic neural networks for hierarchical reinforcement learning (SNN4HRL) (Florensa et. al., 2017)[3]. However, the option-critic architecture does not seem to learn useful or meaningful options, while SNN4HRL assumes there are existings skills. Can you design an approach to learn and reuse useful options while solving tasks? A Useful option may mean it may have a good probability that it will be reuse later.

8 Automating Stock Trading

Can you design a learning algorithm to predict the stock price in a long horizon? You may be just predict whether the next day is up or down. You can get data from `https://www.quandl.com/`.

References

[1] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI*, pages 1726–1734, 2017.

[2] Michael Bowling. Convergence and no-regret in multiagent learning. In *Advances in neural information processing systems*, pages 209–216, 2005.

[3] Carlos Florensa, Yan Duan, and Pieter Abbeel. Stochastic neural networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1704.03012*, 2017.

[4] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic

motivation. In *Advances in neural information processing systems*, pages 3675–3683, 2016.

[5] Tor Lattimore and Marcus Hutter. Pac bounds for discounted mdps. In *International Conference on Algorithmic Learning Theory*, pages 320–334. Springer, 2012.

[6] Ofir Nachum, Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *arXiv preprint arXiv:1805.08296*, 2018.

[7] Satinder Singh, Michael Kearns, and Yishay Mansour. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 541–548. Morgan Kaufmann Publishers Inc., 2000.

[8] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks. *arXiv preprint arXiv:1804.00645*, 2018.

[9] István Szita and Csaba Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1031–1038, 2010.

[10] Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems*, pages 2154–2162, 2016.

[11] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1703.01161*, 2017.

[12] Chongjie Zhang and Victor R Lesser. Multi-agent learning with policy prediction. In *AAAI*, 2010.

[13] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.