CS 4180/5180: Reinforcement Learning and Sequential Decision Making (Spring 2025) – Final Project

1 General Information

The goal of this project is to apply concepts/techniques learned in class to a substantial problem that interests students.

Topics

Your final project can be on anything that has some relation to RL and/or sequential decision making. There are two types of projects:

- **Practical**: Find and formulate a problem, develop/implement/apply algorithms to solve it. Recommended if you want to escape the toy domains we have studied so far and actually build something substantial and potentially applicable in the real world. You would likely learn how to use new RL-related software tools and hone algorithmic / data-scientific skills. If successful, the project would be a great addition to your portfolio. Also a great chance to do the RL-related hobby project that you have always dreamed of but never got started on.
- **Theoretical**: Do an independent study of an advanced topic not covered in the course. Recommended if you have enjoyed the topics so far and just want to see more of it, or go significantly deeper into something we briefly touched on. You will learn about new problems / models / algorithms. Could also be used to reproduce and evaluate an approach proposed in a research paper you've read. In this case, you should propose and evaluate variations on the ideas from the paper. These don't need to be major variations, but you should expand upon the ideas in the paper in some way.

Teams

Teams should have 1–2 members. Teams with 3 members are possible, but should have larger scope. Please talk to the instructor before proposing a threee person project.

Expectations

- Practical project: Your team is expected to find a problem of interest, formulate it well and precisely, develop/apply an algorithm to solve the problem, analyze your results, and communicate that in the presentation and the final report. Your experiments should compare at least two different closely related algorithms or variations on the algorithm. Make sure the methods being compared are related; this should not be a comparison between totally different methods.
- Theoretical project: You should propose a new algorithm or study variations on an existing algorithm that we have not studied in class. You should describe your method clearly and analytically. Evaluate your method versus baselines (i.e. standard approaches) in the literature or that we have studied in class. Evaluate the importance of various aspects of your method by performing ablation studies, i.e. evaluate the importance of specific elements by creating versions of the method that omit those elements and comparing against the full version of the algorithm.

Typical project process

Overall, we expect each team member to spend about 40–60 hours on the project.

1. Decide on a project topic, do some background reading, and make an initial problem formulation. Make a rough plan for the project itself (this is the proposal).
2. Learn the additional knowledge necessary to undertake the project. The amount of time this takes may vary depending on your chosen topic. It is very likely that you will need to learn at least one or two new concepts/algorithms for your project.
3. Figure out the simplest way to solve your problem (i.e., a basic algorithm), and implement it / use an existing implementation. Apply the simple algorithm to your problem.
4. There are usually two results of step 3: Either you have completely solved the problem you chose, or it doesn't work too well (at least on some cases). If the former, analyze why it did so well, and come up with some interesting extensions. If the latter, analyze what went wrong, and figure out how to address that.
5. Iterate steps 3-4, each time using the analysis in step 4 to identify some potential area to improve on, and try again (step 3).
6. Summarize the overall process: the problem, algorithm, results, and insightful analyses. Present your story to the class / in your final report.

2 Potential Topics

This section provides some general topic categories and specific suggestions, but you are welcome to choose anything else that interests you.

Topic categories

- (Practical) Find a decision-making problem in real life that is worth automating. When we consider agents, the first thing that comes to mind may be game-playing agents and robots, which are certainly great applications. However, many other things can (and do) use automated decision making, e.g., elevator scheduling, transportation systems (traffic lights, lanes, etc.), electric grids, logistics and warehousing, computer processor and memory control, etc. RL has actually been applied to many of these; go explore.
- (Practical) OpenAI Gym – This platform from OpenAI contains many interesting environments for training reinforcement learning (RL) algorithms. It is quite popular for RL enthusiasts and is often used to benchmark RL algorithms. DeepMind Lab is a another such platform, but the environment may be more challenging (first-person 3-D) – and potentially more interesting. There is an increasing number of similar platforms. Note that it is insufficient to simply apply an existing codebase (e.g., DQN) on an existing benchmark domain (e.g., Atari) – you should implement your own algorithms and/or extend existing methods.
- (Theoretical) Learn extensively about a problem setting, model, or algorithm that is just beyond the scope of the course. Implement that method on an approprately challenging problem and evaluate variations of the idea that you think would be interesting.

- (Theoretical) Find a paper that tackles an interesting problem, and try to re-implement it (initially without referring to existing code, if any is available). This can be quite challenging because you may find that some crucial details (e.g., particular settings of constants or hyperparameters) may not be present in the paper! Propose and evaluate variations on the method that you think might be interesting.

Potential topic areas not covered in the course (non-exhaustive)

- More on deep reinforcement learning (see recent papers at ICLR / ICML / NeurIPS / etc.)
- Robot learning (see recent papers at CoRL, RSS, ICRA, IROS, etc.)
- RL in X (e.g., healthcare, education, logistics, NLP, games, robotics)
- Imitation learning / learning from demonstration
- Temporal abstraction / state abstraction in MDPs and RL
- Exploration / intrinsic motivation / curiosity / curriculum learing / safety in RL
- Multi-agent RL / Bayesian RL / Partially observable RL / POMDPs

Projects that are *not* recommended

- Apply an existing algorithm from an existing codebase to an existing domain/benchmark and report the results. This is not enough – you need to analyze the results, see what was good and what could be improved, and iterate.
- Projects on unrelated topics – if you are unsure, ask us!
- Projects that are too broad. You do not have a lot of time. Start small (very small), and if you succeed early, extend and iterate from there. If there is an interesting problem that is likely to take too much time (this is true for many interesting problems), identify the first step in the problem and make that your project.

Comments regarding implementation (code)

- All projects should evaluate a method of some kind in code. You can use whatever programming language you like. This choice may be dependent on any existing code/libraries that you build upon.
- If there is an existing implementation, you can use it (with proper acknowledgment), but you do not have to. It depends on what you want to get out of the project. If you want to extend an existing method, then you will probably save time by building on existing implementations (especially if this is some non-RL related infrastructure, e.g., a game engine). If you want to have the experience of writing from scratch and deeply understanding the existing approach, you are welcome to do a re-implementation (although you should not refer to the existing "answer" in this case in your initial attempt). If you choose to re-implement, you may find that you cannot replicate existing performance, at which point you can compare against the existing implementation and see what went wrong – is it a bug on your end, or are there some 'magic numbers' that make the algorithm work? Analyzing these hidden / unexpected bits can make a very interesting project.

- Keep in mind that RL methods take a while to run (recall experience in assignments), and modern deep RL methods take a long time to run (e.g., on the order of multiple days is not uncommon, even with very high-end CPUs and GPUs). Make sure you have a plan to start small and iterate quickly. If you do plan on eventually scaling up, make sure you have access to high-end computing resources, e.g., your own server, the Discovery cluster, cloud computing, etc. Note that we will not be able to provide technical/financial support for any of these external resources, so make sure you are comfortable using them early on. There are many RL projects that do not require such intensive computing resources.

3 Project Proposal

Each team submits one proposal. The project proposal is a short document that organizes your team's intentions and communicates that to the course staff, such that we can provide appropriate guidance. It is not a contract – we expect that projects may change course after you start working on them. As such, the proposal will not be graded; however, if you do not submit a proposal, it may negatively affect your project grade. There is no specific page limit, but we expect that a 1-2 pages (12-pt single spacing) should suffice. Just make sure to include all the following items:

- Who is on the team? If you are working in a team of two, is there a clear division of labor? If team is smaller/larger than recommended, provide a rationale.
- Describe the problem you are trying to address, and provide a formulation of it. If it involves an algorithm (i.e., any implementation-based project), describe the input to the algorithm and the desired output.
- What is the ideal outcome of the project? What do you expect to show?
- What algorithms do you expect to use?
- What topics / libraries / platforms, if any, will you have to learn in order to undertake your project? Provide references where applicable.
- What domain(s) will you be working on? Is there a simulator available? If not, will you making a new one? If the latter, do you have the resources / data to do so?
- Identify the comparisons and/or ablation studies you plan to do in the project. You must compare two or more different algorithms or two or more different variations on the same algorithm. Describe what you plan to do and why this would be an interesting comparison to perform.
- Provide a week-by-week plan for your project.

4 Final Report

There is no specific format that the project report should follow. You should choose a structure that best tells the story of your project. Most reports will probably have these components:

- Describe the problem on a high level, including some motivation for choosing it (just like in your presentation).

- Concrete technical problem statement, model (if applicable), inputs/outputs.
- Describe the dataset / simulator you used, if applicable.
- Methods / algorithms that you used or developed. If you are using algorithms not covered in the class, provide a description of it so we know you understand what you have been using.
- If you are basing your project on someone else's work, explain on a coarse level what they have done, and if you are doing anything different.
- Empirical results, including details on the experimental setup (be precise about what settings / data you ran experiments with). What were your hypotheses / what do you expect to see from your experiments?
- Analysis of empirical results. What worked, what did not, and why? If this is a practical project, make sure there is a comparison between at least two different variations of the method.
- Discussion / future directions (if any). What difficulties did you encounter, if any? Did anything not go according to plan? If you had more time to spend on the project, what would you have liked to do next?

Length

There is no set length / page limit for the report. It would probably take at least 6–8 pages (including figures) to include all of the above. You can write as much as you wish, but do not be excessive – just get to the point and provide a complete description of the project, including the components described above.