Machine Learning Project 2018-2019
Exploring Multi-Agent Learning

Karl Tuyls, Wannes Meert

February 2018

Read this text completely. Deviations from the instructions may result in lower scores.

1 Introduction

We live in a multi-agent world and to be successful in that world, agents, and in particular, artificially intelligent agents, need to learn to take into account the agency of others. They will need to compete in market places, cooperate in teams, communicate with others, coordinate their plans, and negotiate outcomes. Examples include self-driving cars interacting in traffic, personal assistants acting on behalf of humans and negotiating with other agents, swarms of unmanned aerial vehicles, financial trading systems, and robotic teams.

This assignment covers some topics in multi-agent reinforcement learning (RL). We will assume elementary knowledge of single-agent reinforcement learning (a good introduction is available at `http://incompleteideas.net/book/the-book-2nd.html`.)
When moving from single-agent RL to multi-agent RL, Game Theory plays an important role as it is a theory of interactive decision making. Throughout the assignment you will use some elementary game theoretic concepts [13] (see `http://www.masfoundations.org/mas.pdf`) in combination with multi-agent learning, which is non-stationary and reflects a moving target problem (for some references see [4, 17, 2]). In this assignment we start by tackling some benchmark games from the 'pre-deep Learning' period (for some Deep RL references see [10, 6]), after which we transition to more complex systems and delve deeper into deep multi-agent reinforcement learning.

Note that we will be organizing a number of introductory sessions (including Q&A) that will cover some of these game theoretic and multi-agent learning concepts. It is highly recommended to attend these sessions.

2 Problem

In this assignment you will be using machine learning (reinforcement learning) and game theory techniques in the context of multi-agent settings. We will focus on zero-sum games and social dilemmas: zero-sum games correspond to situations in which the total sum of gains and losses of all players involved equal to zero; and social dilemmas are known as games in which several agents participate and there is a tension between what is optimal to do at the individual level (in the short term) and what is beneficial to do at the group-level (in the long term). Famous examples of the former include the *Rock-Paper-Scissors* game and *Poker*, and of the latter include the *Prisoner's Dilemma* game and the *Tragedy of the Commons*. In this assignment we

will be particularly interested in what are called commons dilemmas, i.e., situations in which an individual is tempted by a personal benefit, but depleting a resource that is shared by all.

Therefore the following approaches become very relevant for agents to deploy in order to deal with such situations:

1. **Reinforcement Learning**: Reinforcement Learning (RL) [1, 15, 16] can be used to learn a policy for playing games. Often, given the very large number of state-action combinations, standard model-free techniques such as Q-learning, with a table of state-action pairs will not work; instead some kind of generalization is needed (predicting the Q-value of unseen state-action pairs), often done using neural networks. There will be a need to design or learn features describing states and actions that correlate with the quality of state-action pairs.

2. **Game Theory**: Game Theory is a theory of interactive decision making, in which the players or agents involved take actions of which the success depends on the actions and preferences of the other players involved. The most elementary concepts include normal form games (an abstract model of a game) and Nash equilibrium. Intuitively, a Nash equilibrium is a strategy profile (a tuple in which each player plays one strategy) in which none of the players have an incentive to unilaterally deviate from their strategy given that the other players keep their strategy fixed. This means they cannot improve their payoff or reward by changing their strategy when the other players keep their strategy fixed.

For both approaches you will need to make implementations of your own, i.e. in RL, you will need to represent states and state-action pairs. In simple games this will be stateless, for more complex settings you will need to use models that can generalize. To this end, you will have to think about features that can accurately represent the game state and that allow for learning models that generalize well. The features can be designed by an expert, or learned automatically. The Q or V function will be expressed in terms of these features. Learning this function can be done using almost any of the machine learning techniques you have seen in the machine learning course: k-nearest neighbors, random forests, neural networks, inductive logic programming, etc. In this assignment we invite you to study Deep Learning for this purpose. Among deep learning methods, convolutional neural networks (CNNs) and Actor-Critic methods like A3C are the most popular [12, 11, 14]. In Game Theory you will need to implement simple games and strategic tools such as Nash and replicator equations. For the final part of the assignment, you will investigate a more complex tragedy of the commons game, i.e. the apple foraging environment or Harvest game. A template implementation will be provided to you.

Figure 1: Gathering Apples Game.

3 Approach

Your task is to investigate and implement multi-agent learning approaches in elementary games and subsequently in the provided apple foraging environment. Additionally, the goal is to study the behaviour of the learning algorithms in game theoretic terms: do the outcomes form a Nash Equilibrium? Is the equilibrium Pareto optimal? How does the behaviour evolve over time and are agents behaving in a fair manner? (metrics that could be of interest are collective return, apple consumption, and sustainability)
The final goal is to implement an agent according to the inequity averse model described in [5, 9]. Are such inequity averse agents capable of cooperation and sustainability in the Harvest game, and not overgraze the world in which they live, i.e., can they be successful in a commons dilemma? This agent should be able to connect to a simple game application using websockets. An agent template, a web-based GUI application, a script to automatically play the game, and technical instructions are available at `https://github.com/ML-KULeuven/the_apples_game/`. Corrections and improvements to the game application are welcome and contribute to a positive evaluation (it is not allowed to share your machine learning model or algorithm with others).
You will work on this project in a team of two or three students.

4 Tasks

The assigment is divided in three tasks, which are described below.

4.1 Form Groups

Before Feb 23rd, 23:59

Mail to both wannes.meert@cs.kuleuven.be and karl.tuyls@kuleuven.be whether you work on this project in a team of two or three and include the names of all team members (one email per team suffices).

4.2 Report 1

Before March 16th, 23:59

In a first phase of the project you are expected to concisely report about the first steps of the project that are outlined below. Mail a report (PDF, $\leq$3 pages) to the aforementioned email addresses.

Literature:

- Describe what **literature** you have read and what you have learned from it. This is not just an annotated bibliography, but a critical analysis of the existing work and how your project relates to it.[1]

Task 1: warming up 'Learning & Dynamics'

- [**Independent learning in benchmark matrix games**] In a first step the purpose is to implement one or more basic reinforcement learning algorithms (e.g. Q-learning, Learning Automata) and experiment with these in two benchmark matrix games that are multi-agent: (1) the 2-player prisoner's dilemma and (2) matching pennies games. Each of these games represent a category of game, i.e. social dilemma and a zero-sum game. Here you need to investigate and report on whether the learning algorithms converge to Nash equilibria, and whether these are (Pareto) optimal? (note that both agents should be using an RL algorithm; if they use the same RL algorithm this is called self-play)

[1] `http://www.writing.utoronto.ca/advice/specific-types-of-writing/literature-review`

- **[Dynamics of learning in benchmark matrix games]** Implement the basic form of replicator equations (selection mechanism only) and examine their directional field plots for the same games and compare these to the learning trajectories. Provide trajectory and directional field or phase plots to illustrate your work. Figure 2a illustrates an example phase plot of replicator dynamics in the matching pennies game, with the Nash equilibrium at $(0.5, 0.5)$.
- Draw some overall conclusions about both steps relating the observed learning and dynamics behaviours in the two benchmark games.

Tasks 2 & 3. Additionally, we also expect you to describe briefly how you plan to address tasks 2 and 3. This part will not be graded, and is only used to give feedback.
[*With task 1 you can earn 6 out of 20 points of your overall mark.*]

4.3 Report 2

Before May 13th, 23:59

Submit your report (PDF, ≤10 pages) and prototype to wannes.meert@cs.kuleuven.be using the Belnet Filesender.[2] You will get an automated notification from the Filesender service when we receive your submission. Your report should fulfill the following criteria:

- Summarize your solution in an **abstract**.
- Clearly state the **problem statement**.
- Formulate your design choices as **research questions** and answer them.
- Write out the **conclusions** you draw from your experiments together with a scientifically supported motivation for these conclusions.
- Be concrete and precise about methods, formulas and numbers throughout the text. A scientific text should allow for **reproducibility**.
- Clearly **cite** all your sources.
- Report the total **time each of you spent** on the project, and how it was divided over the different tasks mentioned.

Task 2: Opponent modelling In a second phase we are going to explore the zero-sum Rock-Paper-Scissors game and opponent modelling using fictitious play.

The goal is now to explore how opponent modelling might be important to find better solutions in multi-agent learning. A famous way to do this is called *fictitious play* [3, 7, 8], another method from Game Theory.

- In a first step we would like you to implement and experiment with Q-learning in the iterated Rock Paper Scissors game – can you visualize your results (e.g. learning traces and dynamics trajectories)?
- Now we would like you to implement the fictitious play method for the Rock-Paper Scissors game – can you visualize your results?
- Finally, the purpose is to combine Q-learning with opponent modelling – can you visualize your results?

[*With task 2 you can earn 7 out of 20 points of your overall mark.*]

[2] https://filesender.belnet.be

Task 3: Harvest Game Agent As a final part of this project we would like you to tackle the Harvest game.

- Start by reading the Appendix (A) of this document which describes the Harvest game.
- Implement your own agent to play the game. Start from the code available at `https://github.com/ML-KULeuven/the_apples_game`
- In the report, motivate and discuss the methods you use:
  - Which (deep) learning algorithm for game playing do you use?
  - What settings of the Harvest game did you examine?
  - How did you evaluate your solution?
- Include all the code you used to perform the machine learning and experiments with the final report (also if the final application runs in the cloud). Running and experimenting your application is part of the evaluation. You are free to use any mainstream programming language and any publicly available toolbox. Make sure that:
  - The code can be executed from the command line using: `./agent 8088` (or `agent.cmd 8088`) and opens a websocket on the given port that understands the commands used by the game application.
  - The default setting is to use the model and parameters you have found to perform the best.
  - Your code is self-contained and easy to setup. It includes all non-standard dependencies and/or includes a dependencies file for a standard package manager (e.g. requirements.txt, setup.py, package.json, Cargo.toml).

[*With task 3 you can earn 7 out of 20 points of your overall mark.*]

4.4 Peer assessment

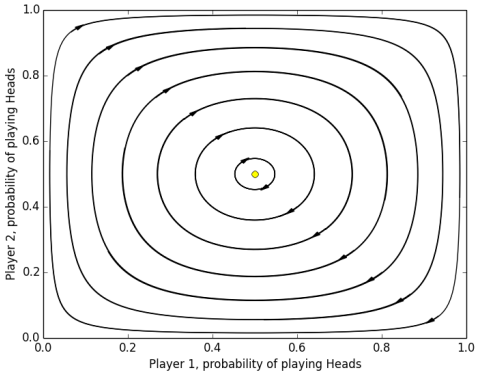
Before May 13th, 23:59, individually

Send by email a peer-assessment of your partner's efforts. This should be done on a scale from 0-4 where 0 means "I did all the work", 2 means "I and my partner did about the same effort", and 4 means "My partner did all the work". Add a short motivation to clarify your score. This information is used only by the professor and his assistants and is not communicated further.

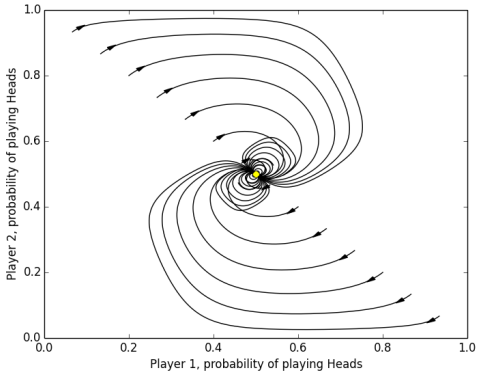
[*The final mark per task is determined by the combination of the reports and the oral discussion*]

A Harvest Game

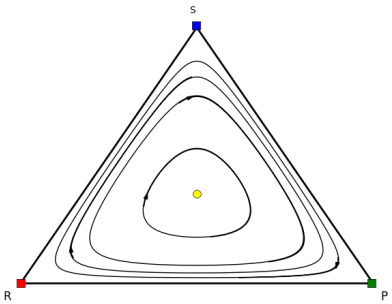
The purpose of the Harvest game is to collect apples. Each apple provides a reward of 1. The apple regrowth rate varies across the map, dependent on the spatial configuration of uncollected apples: the more nearby apples, the higher the local regrowth rate. If all apples in a local area are harvested then none ever grow back. After 1000 steps the episode ends, at which point some metrics can be computed, and the game resets to an initial state. The dilemma is as follows. The short-term interests of each individual leads toward harvesting as rapidly as possible. However, the long-term interests of the group as a whole are advanced if individuals refrain from doing so, especially when many agents are in the same local region. Such situations are precarious because the more harvesting agents there are, the greater the chance of permanently depleting the local resources. Cooperators must abstain from a personal benefit for the good of the group.



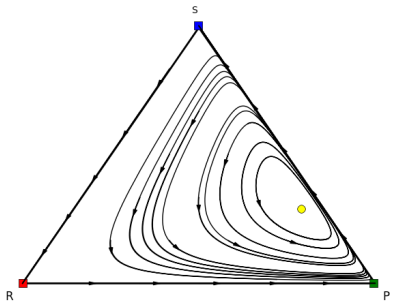
(a) Evolutionary dynamics of the Matching Pennies game.



(b) Average Time Evolutionary dynamics of the Matching Pennies game.



(c) Evolutionary dynamics of the Rock-Paper-Scissors game.



(d) Evolutionary dynamics of the biased Rock-Paper-Scissors game.

Figure 2

The total size of the playable area is 36×16 for Harvest. Games are partially observable in that agents can only observe via a 15×15 window, centered on their current location. The action space consists of moving forward, turning left and right. Optionally you can add the ability to tag each other. This action has a reward cost of 1 to use, and causes the player tagged to lose 50 reward points , thus allowing for the possibility of punishing free-riders. The game is played by minimally 5 agents (and we suggest to not choose more than 15).

References

[1] Hendrik Blockeel. *Machine Learning and Inductive Inference (course notes)*. KU Leuven, 2017.

[2] Daan Bloembergen, Karl Tuyls, Daniel Hennes, and Michael Kaisers. Evolutionary dynamics of multi-agent learning: A survey. *J. Artif. Intell. Res. (JAIR)*, 53:659–697, 2015.

[3] George W. Brown. Iterative solution of games by fictitious play. In T. C. Koopmans, editor, *Activity Analysis of Production and Allocation*. Wiley, New York, 1951.

[4] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 38(2):156–172, 2008.

[5] Steven de Jong, Simon Uyttendaele, and Karl Tuyls. Learning to reach agreement in a continuous ultimatum game. *J. Artif. Intell. Res.*, 33:551–574, 2008.

[6] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

[7] Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 805–813, 2015.

[8] Josef Hofbauer and William H. Sandholm. On the global convergence of stochastic fictitious play. *Econometrica*, 70(6):2265–2294, 2002.

[9] Edward Hughes, Joel Z Leibo, Matthew Phillips, Karl Tuyls, Edgar Dueñez Guzman, Antonio García Castañeda, Iain Dunning, Tina Zhu, Kevin McKee, Raphael Koster, Heather Roff, and Thore Graepel. Inequity aversion improves cooperation in intertemporal social dilemmas. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 3330–3340, 2018.

[10] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[11] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016.

[12] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[13] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[14] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[15] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 2nd edition, 2017.

[16] Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Morgan & Claypool, 2010.

[17] Karl Tuyls and Gerhard Weiss. Multiagent learning: Basics, challenges, and prospects. *AI Magazine*, 33(3):41–52, 2012.