# LECTURE 1

*Introduction to Reinforcement Learning*

This is an introductory lecture designed to introduce people from outside of Artificial Intelligence to the Reinforcement Learning problem.

NOTE: In the spirit of syllabus week, this lecture is significantly slower than most lectures in the course.

NOTE: These notes only concern a portion of the material covered in class, and at times do not involve a topic's in-depth discussion.

Table of Contents:



## Administrativa

See the syllabus at [cmsc389f.umd.edu](cmsc389f.umd.edu).

## What is Reinforcement Learning?

Reinforcement Learning is an area of AI that utilizes the concept of learning through interacting with a surrounding environment.

Think about teaching a dog a new trick. You can't tell it exactly what to do, but you can give it a treat (a reward) for doing the right actions. The dog uses these treats to figure out what the right actions to take are, with the goal of maximizing its total number of treats (cumulative reward). This is the essence of the Reinforcement Learning problem: interacting with the environment (what is around you) to maximize reward. We will see later in this course that we can formulate *any* goal as the maximization of some reward.

Reinforcement Learning is one of the *hottest areas* in tech right now. It was named one of MIT Technology Review's "10 Breakthrough Technologies of 2017". It is also the main driver of innovation behind industry titans such as DeepMind, OpenAI, and Tesla.

**How does RL compare to other areas of ML?**

The most popular area of ML nowadays is supervised learning. With supervised learning, the goal is to learn a model (a function) to accurately classify data into categories.  For example, we have 100 cats  and 100 dogs, we have to separate them into categories. As a human, we use our internal model, created from the hundreds of cats and dogs we've seen in the past, to differentiate between cats and dogs, based on certain "features" of a dog or cat (ex: does it have whiskers, or does it bark).

To learn this function, we "train" our function/model using labeled data that is already categorized into its correct categories. It's all about discovering structure (relationships between features and the final classification) inside of a dataset.

Compared to supervised learning, reinforcement learning does not use labels, but instead uses rewards, which can be thought of as delayed labels. With RL, we do not know how good a sequence of actions was until later, when we can see our total reward. Whereas with Supervised Learning, we immediately know the label of new input data by running it through our function we have learned.

The Reinforcement Learning problem is more difficult compared to Supervised Learning. It requires more knowledge of the problem, such as knowing the possible actions to take or information that can help us evaluate how good it is to be in a particular state (ex: how good is it to be in the flag state of Super Mario vs getting hit by Goomba state?)

That is why nowadays Reinforcement Learning is often used in simulated environments, such as video games, where we can create multiple simulations and collect vast amounts of information easily.

So why use RL? It is very useful in scenarios where we do not know the optimal sequence of actions to take. If we don't know the best actions to take, then we can't use labels to train our model. But with Reinforcement Learning, we can figure out the best actions to take, given that we have a criterion to evaluate how good our actions were (a reward).

Ex: Monitoring temperature of server systems (real world Google example, saving millions of dollars)

# History of Reinforcement Learning

"Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The great the satisfaction or discomfort, the greater the strengthening or weakening of the bond." - (Thorndike, 1911, p. 244)

The above quote is by the author of Modern Educational Psychology, and is related to the biological inspiration that has led to many of the guiding principles of reinforcement learning.

Reinforcement Learning mainly came about as a result of the contributions of these two fields*:

1. Optimal control (from engineering)
2. Trial-and-error learning (from biology/psychology)

* *not entirely accurate, but sufficient for the topics covered in this course*

Optimal control is an area of engineering that deals with finding a way to achieve some optimality criterion in a system. As expected from a field with engineering roots, it has been rigorously mathematically defined since its inception. In this area, we assume we have almost *omniscient* knowledge of the system.

**Example**: Say Jim is driving back from I-270 after a long day of classes, and he wants to get home as fast as possible. An optimal control problem could be "How much should Jim accelerate to get home as fast as possible?". The system here would consist of Jim and the road, and the optimality criterion would be the minimization of the Jim's travel time, under some obvious constraints such as the amount of gas in Jim's car, the local speed limits, etc.

Trial-and-error learning is an area that was first studied in psychology (see Pavlovian conditioning), dealing with the way in which humans and animals learn through experimentation, observation, and subsequent behavioral updates. In this area, the environment *does not change* with respect to the animal's actions. In

contrast to optimal control, quantitative models were largely absent from this field until quite recently.

**Example**: 5-year-old Jim walks into the kitchen. Little Jim sees a glowing red circle on the stove. Little Jim reaches out his hand and touches it. Ouch, that hurt! Little Jim decides to never touch the red-hot stove ever again.

Using the real-world observations of animal learning that we garnered from psychology, we distill the fundamentals of a learning system into two concepts: an agent, and an environment. By codifying this into a rigorous, mathematical framework like engineers have done in optimal control for years, we arrive at a system that we call Reinforcement Learning, where aspects of the system can be *unknown* and the environment *can change* in response to agent actions. The result is a system more general than optimal control, and more powerful than cognitive learning.

We will continue this practice of borrowing concepts from areas far older than Reinforcement Learning throughout the rest of this course.

Why study RL now?

- Computational resources (big data, powerful GPUs/TPUs)
- Deep Learning
- Advances in Reinforcement Learning

# Agent-environment framework

There are two key parts of an Reinforcement Learning system:

- The agent
- The environment

The agent is what is being controlled by us. Agents take actions within an environment, which in turn responds to the agent's actions.

The environment can be represented as a set of states that the agent can exist in, with transitions between each state. (For example, imagine a student in a "Life" environment. Possible states can be "sleeping", "eating", and "going to school". From "sleeping", if the student "wakes up", they can move on to something like the "eating state".

The goal of Reinforcement Learning is to figure out actions we can get the agent to take in the environment, in order to maximize some cumulative reward.

**Example:** Think of the classic game of pong.

The paddle is the agent. It can only take two actions:
1. move up,
2. move down.

The environment is the game. It consists of the ball, the opposing paddle, the game physics, and the overall "clock" i.e. some N seconds since the start of the game.

The reward is scoring a point. The goal is to win the game. We can formulate the goal as the agent accumulating as many points as possible.

Notice that if we find some way to tell the paddle agent when to move up/down in order to maximize the number of points it scores, we will have created a Pong AI. This is the essence of reinforcement learning. Now, let's get into the details of what it means to be an agent.

If the paddle tries to take a "move up" action, the game environment will respond by moving the paddle into a new state. This new game state is one where:
1. the paddle is one pixel above where it was before
2. the overall clock is one second later than it was before

Suppose for example, we have a Pong game where the height of the screen is 4 pixels and the size of the paddle is 1 pixel. For simplicity, we will represent the environment state as a tuple: (height, time). The game starts with the paddle at the bottom of the screen. Thus, the paddle will be in the initial state: (height 1, time 0). If the paddle takes the action "move up", the game environment will send the paddle a new state: (height 2, time 1). By continuing in this manner, we will have successfully formalized playing a Pong game.

Notice that there are some subtleties involving constraints that the environment will need to process (e.g if the paddle is at the bottom of the screen, the environment should move the paddle into a state

where the time has increased by one, but where its position has not changed). Later in this class, we will see that we can model *any* real-world environment using a concept of a collection of states and actions.

**Additional Example:** Your agent is a robot, shown below. All the possible tiles the robot can be in are all the possible states of the environment. The robot can move in all four directions (up, down, left, right). The robot moving in a direction (ex: moving up) results in an environmental state change, since the tile the robot is currently on now changes in response to moving.

Thus far we have only discussed Reinforcement Learning as a problem and we have not discussed any *solutions* to the problem. We need to spend some time to formalize the system before we even begin to formulate solutions, but we will see 4-5 lectures from now that the solution comes easily and intuitively after designing the framework rigorously.