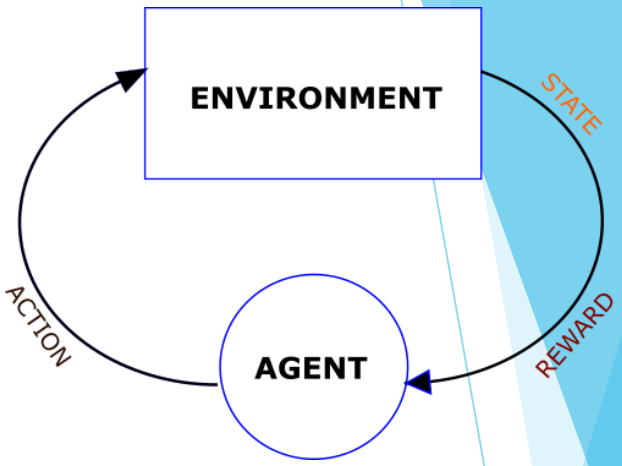
Challenges in the Verification of Reinforcement Learning Algorithms

Adam Budai

SWVV – homework

Reinforcement Learning

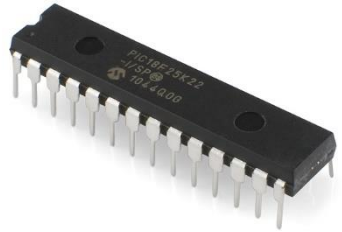


Reward formulation:

- Energy consumption
- Flight time without accident
- Travelling time

video input

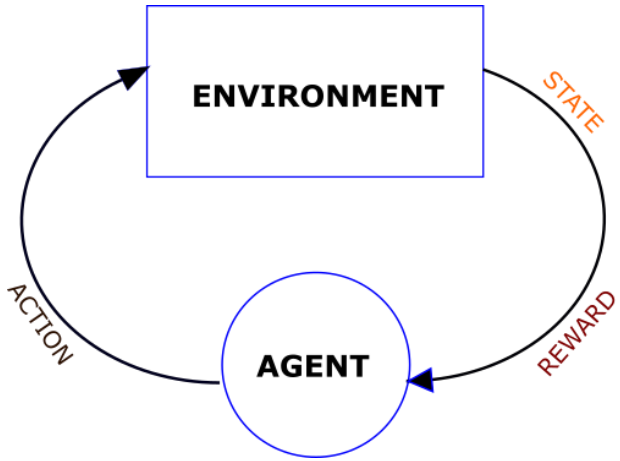
camera



control decision

Reinforcement Learning

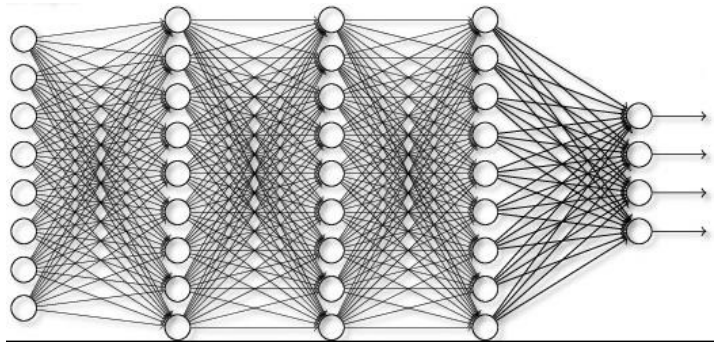
- State: a representation of the environment
- Action: intervention to the env.
- Reward: feed back to the agent (from the env.)



- Policy: (state -> choose an action)
- Goal of RL: find a policy which ensures the **maximum reward** gathered in the long run

Reinforcement Learning

Result



π

Training process

RL algorithm

Assumptions

- Avoiding unexpected violations
- Four categories:
  - Operating environment
  - Platform
  - Assumptions on data
  - Algorithm

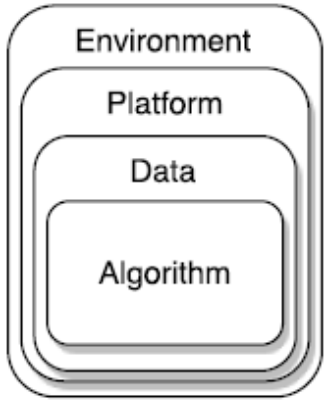


Figure 1. Assumption hierarchy

Environmental assumptions

- Physical environment

Platform assumptions

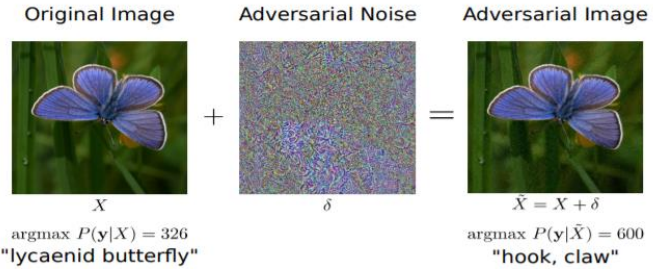
- A system can have more **components**
  - A component can fail (failure assumptions)
    - Usually states the expected time until failure of a component
    - *Redundant* sensors -> could be assumed the sensor data is reliable from the software's point of view (*simplification* for ML)
  - A component's quality of service can change (fidelity assumptions)
    - Data gathered by sensors (noisy, can fail time to time)
    - A ML agent is defined by the data that it was trained on
    - Fidelity of the sensors has impact on ALL of the future outputs of the ML sys.

Src: Jaapson blog

Src: BrianAJackson

Assumptions on data

- Data is the cornerstone of an ML system
  - Independent, identically distributed samples (IID) (assumption)
    - Most of the time not true -> in new env. can cause problems
  - Proximity (assumption)
    - Similar input produces similar output
  - Adversarial behaviour (assumption)
    - Attacks from outside
    - Change the behaviour of the system
    - ML systems will be networked



Src: https://www.etsmtl.ca/Unites-de-recherche/LIVIA/Seminars/Igh_adversarial_examples.pdf

Assumptions on ML algorithms

- the desired functionality can be learned from data
- the chosen algorithm and its structure are able to accurately model the system to be learned
- the algorithm does not get stuck in a local optimum but reaches a global optimum

Interactions between the assumptions

	Environmental assumptions	Platform assumptions	Data assumptions	Algorithm assumptions
Environmental assumptions	×			
Platform assumptions	Platform assumptions can be based on assumptions on the environment, e.g. a sensor has a 95% accuracy at temperatures between x and y.	×		
Data assumptions	The training data is assumed to adhere to the assumptions and constraints of the environment.	The data might be captured by the platform sensors, thus being affected by any fidelity or failure assumptions.	×	
Algorithm assumptions	The chosen algorithm is assumed to model the environment of the system well.	The algorithm used assumes the availability and accuracy of inputs.	Properties of the algorithm such as convergence depend on the data it sees.	×

Table 1. Assumption influence

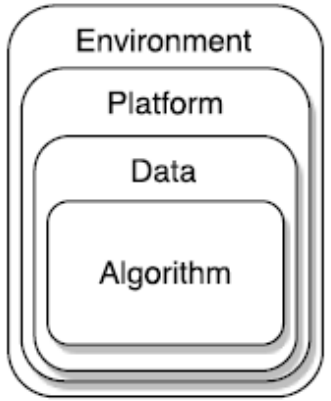


Figure 1. Assumption hierarchy

Verification of RL

- **Formal verification**: mathematically proving that a program meets its specification
- Specification for an ML system is like: (Liang)
  - Input: Training data
  - Output: Weight Vector and an estimated accuracy
- It is not clear how to formalize this -> deal with only critical components of an ML algo. (instead of the entire on)
- E.g: optimization program of an ML program

Verification of RL

- **Motivation**: Reinforcement learning seems to be the most transparent and best understood machine learning category and has historically been used most in cyber-physical systems. (☺)

- **Safety problem**: During learning the agent has to try out new policies (behaviours) which can be demaging for the environment or the agent itself.Restrictions on the behaviour can make the agent safer but can limit its performance. (predictablity/exploration conflict)

Verification of RL – What can be verified?

- Report proposes two categories:
  - Offline verification (training is finished, model is freezed)
  - Runtime verification (during training)

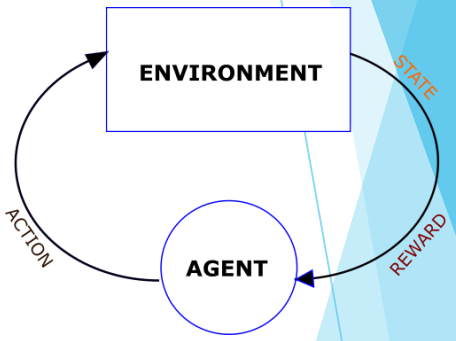
Verification of RL – Offline verification

- State to action mappings (policy)
  - verification of requirements in the form „From all states in this set, never do this action”
- Action sequences
  - If the environment model is known as an MDP, requirements can be verified in some probabilistc temporal logic specification (tool: PRISM)
- Algorithm properties independent of data
  - Like theoretical proofs for convergence (data independent, e.g: Q-learning)
- Validating assumptions on training data
  - After training the whole dataset is available. Statistic parameters can be checked, like variable ranges.

Verification of RL – Runtime verification

- State to action mappings
  - In each step of learning verify it as in the offline case
- Action sequences
  - can be considered as the runtime verification of a temporal logic specification just observing the output actions of the agent
- Monitoring (violations of) assumptions
  - Assumptions made on the environment or the platform can easily be checked at a distinct moment in time, like verifying assumptions on one single instance of training data
- Input instance in training data
  - If an input instance is not in the trainig data, it indicates an unkonwn situation the agent encountered with -> should be avoided

Example



- Automatic gearbox trained with RL
- **State**: speed, rotation per minute, gear
- **Actions**: shifting up, shifting down, staying
- **Goal:** Training the algorithm to achieve the 30 m/s speed as fast as possible
- **Reward:** huge reward if 30 is reached, huge penalty if 30 is not reached within 40 seconds

Example - Assumptions

- AE (Environment Assumption), AP (Platform Assumption), AA (Algorithm Assumption), AD (Data Assumption)

Environmental

- AE1: The gearbox is not used on slopes with a gradient of more than 20%.
- AE2: The controller is only used when the car moves forward, so speed > 0.

Platform

- AP1: The engine and gearbox are working as expected.
- AP2: The engine can handle RPM between 1000 and 6000.
- AP3: The sensor readings have a +/- 5% accuracy.
- AP4: The sensors are assumed not to fail (from a software perspective).

Example - Assumptions

Algorithm assumption

- AA1: Since the algorithm has a formal convergence guarantee, it is assumed to converge.

Data assumptions

- AD1: The data is subject to the proximity assumption.
- AD2: The data is subject to the smoothness assumption.
- AD3: No adversarial forces are working against the algorithm.

Example – safety REQs

- R1: The gearbox controller does not shift down in first gear.
- R2: The gearbox controller does not shift up in fith gear.
- R3: The gearbox controller does not cause the engine to go over 6000 RPM.
- R4: The gearbox controller does not cause the engine to go under 1000 RPM in any gear
- but first.
- R5: The controller should not shift up or down immediately twice in a row.
- R6: The gearbox controller should not shift too quickly.
- R7: Convergence should be guaranteed.
- R8: The rate of convergence should be bounded.

Example – Proposed verification appro.

- Goal is to verify all the safety requirements
- Two methods: offline verification and runtime verification

Offline

- R7-R8 can be proved by the algorithm properties of the used algorithm (Q-learning fulfills these requirements)

- *R7: Convergence should be guaranteed.*
- *R8: The rate of convergence should be bounded.*

Example - Verification

- R1: The gearbox controller does not shift down in first gear.
- R2: The gearbox controller does not shift up in fith gear.
- R1 & R2 state-action mappings
  - In case of deterministic action selection, simply check the requirements
  - If action selection is stochastic, intercept the controller by the verifycation system (force new action instead of the wrong one)

Example - Verification

- R3: The gearbox controller does not cause the engine to go over 6000 RPM.
- R4: The gearbox controller does not cause the engine to go under 1000 RPM in any gear
- R3 & R4 - Probabilistic Action Sequences
  - The model of the environment is necessary
  - If not available learn it
  - Calculate the possible outcomes as a probability distributions
  - Calculate the probabilistic version of the requirements
  - For example: the probability the gearbox causes the engine to go over 6000 RPM is less then 10^{-5}.

Example - Verification

- R5: The controller should not shift up or down immediately twice in a row.
- R6: The gearbox controller should not shift too quickly.
- R5 & R6 - Illegal Sequences
  - The action sequences Up -> Up, Down -> Down, Up -> Down and Down -> Up are illegal.
  - These requirements can be specified in temporal logic and verified at runtime

Example - Validation

- Validity of Environmental Assumptions
  - Easy to do, easy to measure the required values then compare
- Validity of Platform Assumptions
  - Hard, because hard to measure the values
- Validity of Algorithm Assumptions
  - Difficult, theoretical results are necessary
  - Even if theory available the measurements, samplings can cause problems
- Validity of Data Assumptions
  - Not trivial in an online setting for the whole data
  - In offline settings or individual instances it is more easier (bounds, anomalies)

Thank you for your attention