Bayesian optimization - part 1

Hrvoje Stojic

May 24, 2018

The roadmap

The problem

The problem

- What are the hyperparameters and how do we optimize them?

The problem

- What are the hyperparameters and how do we optimize them?
- Some examples:

The problem

- What are the hyperparameters and how do we optimize them?
- Some examples:
  - SVM: regularisation term C, kernel parameters

The problem

- What are the hyperparameters and how do we optimize them?

- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs

The problem

- What are the hyperparameters and how do we optimize them?

- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size

The problem

- What are the hyperparameters and how do we optimize them?

- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)

The problem

- What are the hyperparameters and how do we optimize them?

- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)

The problem

- What are the hyperparameters and how do we optimize them?

- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)

- Standard procedures

The problem

- What are the hyperparameters and how do we optimize them?
- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)
- Standard procedures
  - Grid search

The problem

- What are the hyperparameters and how do we optimize them?
- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)
- Standard procedures
  - Grid search
  - Random Search

The problem

- What are the hyperparameters and how do we optimize them?

- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)

- Standard procedures
  - Grid search
  - Random Search

The problem

- What are the hyperparameters and how do we optimize them?
- Some examples:
  - SVM: regularisation term C, kernel parameters
  - Logistic regression: SGD learning rate, regularization parameter, mini batch size, number of epochs
  - Online Latent Dirichlet Allocation: two learning rate parameters, mini batch size
  - Three-layer convolutional neural network: SGD learning rate, number of epochs, 4 x weight costs (layers and softmax), width, scale and power (the response normalization on the pooling layers)
- Standard procedures
  - Grid search
  - Random Search
- What are (dis)advantages of the usual approaches?

A closer look at the problem

- What is the alternative?

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface

A closer look at the problem

- What is the alternative?

- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful
- Active selection?

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful
- Active selection?
  - Involves balancing exploration and exploitation

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful
- Active selection?
  - Involves balancing exploration and exploitation
  - Strong interaction between the two processes

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful
- Active selection?
  - Involves balancing exploration and exploitation
  - Strong interaction between the two processes
  - Calls for smart selection, probabilistic models make it easier

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful
- Active selection?
  - Involves balancing exploration and exploitation
  - Strong interaction between the two processes
  - Calls for smart selection, probabilistic models make it easier
- When does it make sense?

A closer look at the problem

- What is the alternative?
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next
- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful
- Active selection?
  - Involves balancing exploration and exploitation
  - Strong interaction between the two processes
  - Calls for smart selection, probabilistic models make it easier
- When does it make sense?
  - Optimizing SMBO can be a hard problem

A closer look at the problem

- What is the alternative?

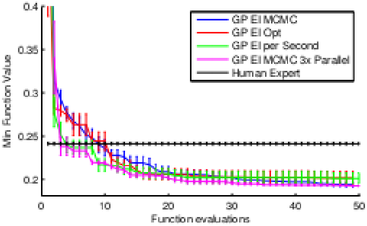
- Sequential model-based optimization (SMBO) algorithms
  - We build a model of the optimization surface
  - Make active choices where to sample next

- Learning a model
  - We can leverage our supervised learning machinery
  - Probabilistic approaches more helpful

- Active selection?
  - Involves balancing exploration and exploitation
  - Strong interaction between the two processes
  - Calls for smart selection, probabilistic models make it easier

- When does it make sense?
  - Optimizing SMBO can be a hard problem
  - Hence, when optimizing costly models, i.e. when time or number of evaluations is very valuable

The main goal: AutoML and hyperparameter tuning

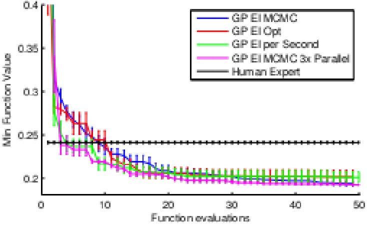


	convex	MRBI
TPE	**14.13** ±0.30 %	**44.55** ±0.44%
GP	16.70 ± 0.32%	47.08 ± 0.44%
Manual	18.63 ± 0.34%	47.39 ± 0.44%
Random	18.97 ± 0.34 %	50.52 ± 0.44%

Table 2: The test set classification error of the best model found by each search algorithm on each problem. Each search algorithm was allowed up to 200 trials. The manual searches used 82 trials for **convex** and 27 trials **MRBI**.

Source: Snoek et al 2012; Bergstra et al 2011

The main goal: AutoML and hyperparameter tuning



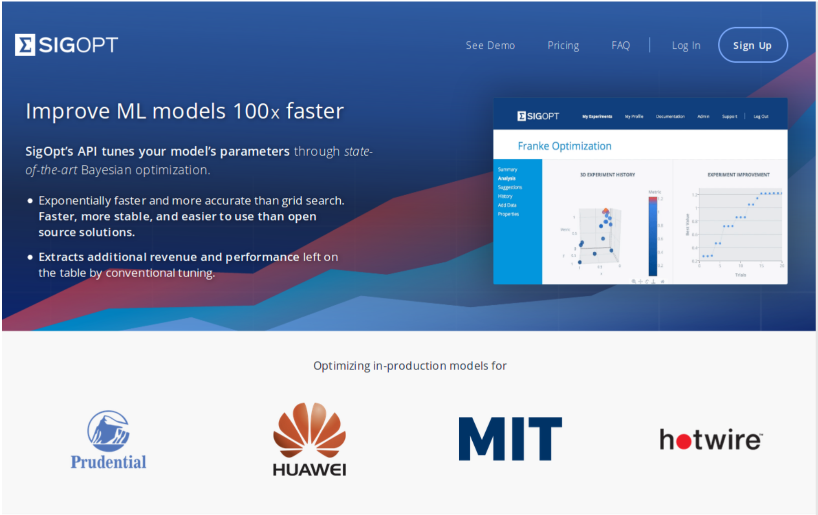
	convex	**MRBI**
TPE	**14.13** ± 0.30 %	**44.55** $\pm 0.44\%$
GP	$16.70 \pm 0.32\%$	$47.08 \pm 0.44\%$
Manual	$18.63 \pm 0.34\%$	$47.39 \pm 0.44\%$
Random	18.97 ± 0.34 %	$50.52 \pm 0.44\%$

Table 2: The test set classification error of the best model found by each search algorithm on each problem. Each search algorithm was allowed up to 200 trials. The manual searches used 82 trials for **convex** and 27 trials **MRBI**.

- CIFAR 10: state of the art was test error of 18%, they achieved 14.98%
- MNIST rotated background images

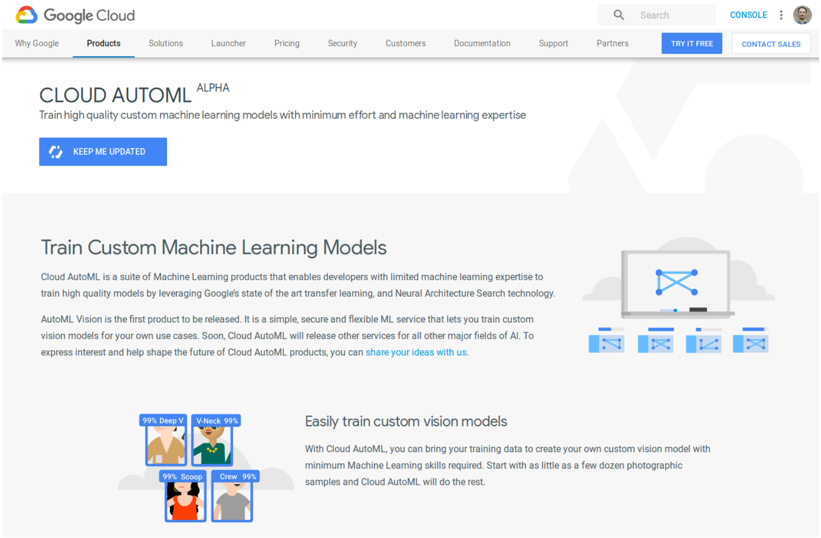
Source: Snoek et al 2012; Bergstra et al 2011

Bayesian optimization going mainstream



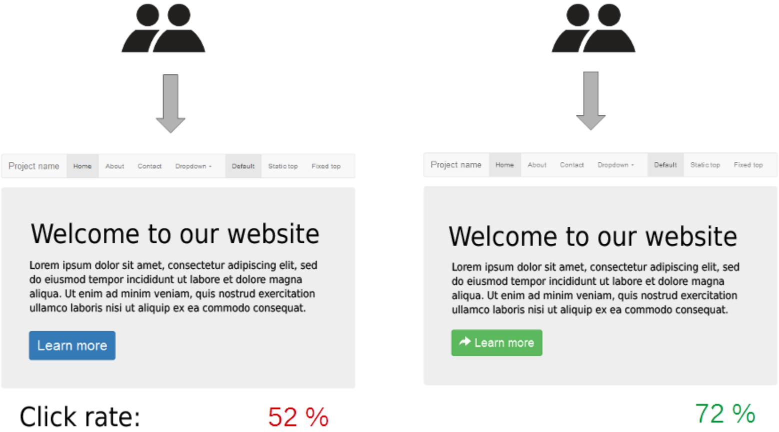
Source: SigOpt webpage

Google Cloud AutoML for computer vision



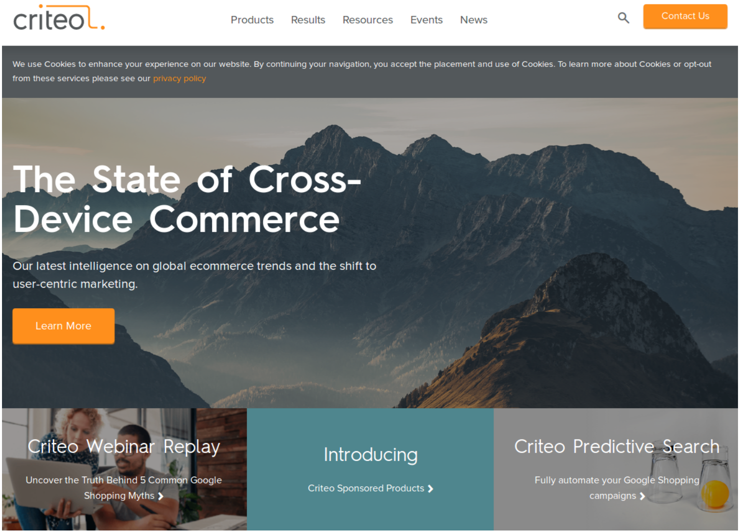
Source: Google AutoML webpage

Bonus - A/B testing



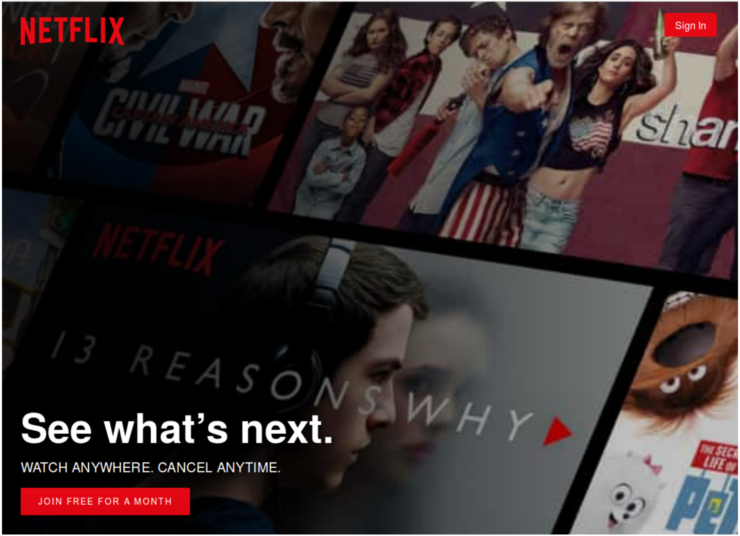
Source: Wikipedia

Bonus - Recommender systems and ad placement



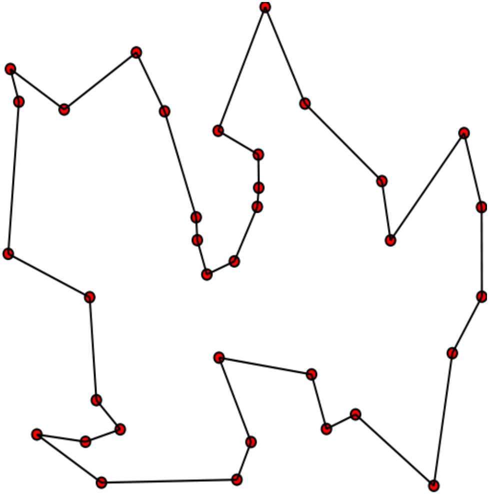
Source: Criteo webpage

Bonus - Preference learning and interactive user interfaces



Source: Netflix webpage

Bonus - Combinatorial optimization



Source: Wikipedia

The roadmap

The roadmap

- Reinforcement learning basics
  - Agents, environments, rewards, states, MDPs
  - Exploration exploitation problem

The roadmap

- Reinforcement learning basics
  - Agents, environments, rewards, states, MDPs
  - Exploration exploitation problem

- MAB problem
  - Classics: ϵ-greedy
  - Frequentist: UCB1
  - Bayesian parametric: Thompson Beta-Bernoulli

The roadmap

- Reinforcement learning basics
  - Agents, environments, rewards, states, MDPs
  - Exploration exploitation problem
- MAB problem
  - Classics: ϵ-greedy
  - Frequentist: UCB1
  - Bayesian parametric: Thompson Beta-Bernoulli
- CMAB problem
  - Frequentist parametric: LinUCB
  - Bayesian non-parametric: GP-UCB

The roadmap

- Reinforcement learning basics
  - Agents, environments, rewards, states, MDPs
  - Exploration exploitation problem
- MAB problem
  - Classics: ϵ-greedy
  - Frequentist: UCB1
  - Bayesian parametric: Thompson Beta-Bernoulli
- CMAB problem
  - Frequentist parametric: LinUCB
  - Bayesian non-parametric: GP-UCB
- Extensions and applications

References

- Reinforcement learning
  - Sutton, R., & Barto, A. (2017). Introduction to Reinforcement Learning (book free of charge: www.incompleteideas.net/sutton/book/the-book.html)
  - D. Silver's lectures (videos and slides: www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html)
- Gaussian Processes
  - Rasmussen, C. E., & Williams, C. K. I. (2006). Gaussian processes for machine learning. MIT Press. (book free of charge: www.gaussianprocess.org/gpml/)
  - Karl Rasmussen's lectures
  - Nando De Freitas' lectures (videos and slides: www.youtube.com/user/ProfNandoDF/videos)

References

- Bayesian optimization

  - Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proceedings of the IEEE, 104(1), 148–175.
  - Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. Advances in Neural Information Processing Systems, 2951-2959.

Software

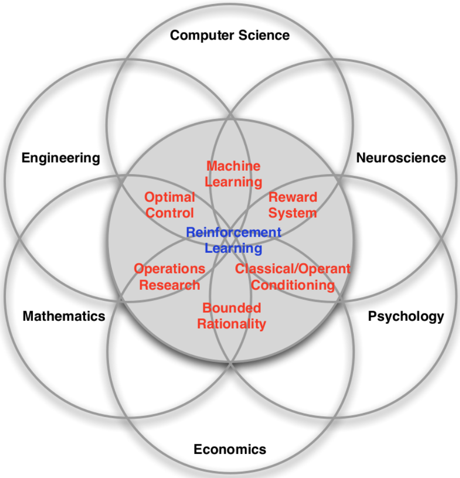
- R packages
  - GPfit, gptk, FastGP
  - rBayesianOptimization (Yan)
  - DiceOptim (Roustant et al., 2012)
- Python libraries
  - scikit-learn, auto-sklearn
  - Hyperopt (Bergstra et al., 2011)
  - Spearmint (Snoek et al., 2014)
- Matlab
  - GPML (Rasmussen)
- C++
  - BayesOpt (Martinez-Cantin, 2014)
- Java
  - SMAC (Hutter et al., 2011)
  - AutoWEKA

Practicalities

- Contact:
  - h.stojic_at_ucl.ac.uk
  - Office hours by video calls
- Evaluation:
  - No exam
  - Individual coding exercise: 40%
  - Group projects: 60%
  - Deadline: June 20

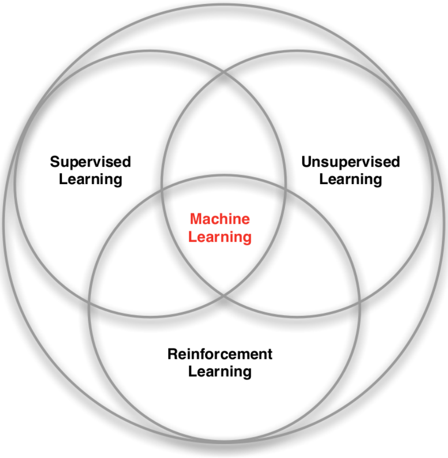
Introduction to Reinforcement Learning

Interdisciplinary area



Source: David Silver lectures

Relation to other types of learning



Source: David Silver lectures

Main characteristics

- Agent receives rewards
  - There is no teaching signal
  - Agent does not observe the counterfactual
  - Goal of the agent is to maximize rewards

Main characteristics

- Agent receives rewards
  - There is no teaching signal
  - Agent does not observe the counterfactual
  - Goal of the agent is to maximize rewards

- Agent has to take actions
  - Exploration exploitation trade off
  - Feedback is (potentially) delayed, credit assignment problem
  - Sacrificing immediate reward to gain more later on
  - Actions (potentially) affect the subsequent data
  - Sequential, non IID data

Main characteristics

- Agent receives rewards
  - There is no teaching signal
  - Agent does not observe the counterfactual
  - Goal of the agent is to maximize rewards
- Agent has to take actions
  - Exploration exploitation trade off
  - Feedback is (potentially) delayed, credit assignment problem
  - Sacrificing immediate reward to gain more later on
  - Actions (potentially) affect the subsequent data
  - Sequential, non IID data
- Examples
  - Robots, autonomous vehicles
  - Managing investment portfolio
  - Optimizing the data centres

Reward hypothesis

- Reward, R_t, is a **scalar** feedback signal
  - Signals how well agent is doing at time t
  - Agent maximizes the long run sum of rewards
  - Exogenously given

Reward hypothesis

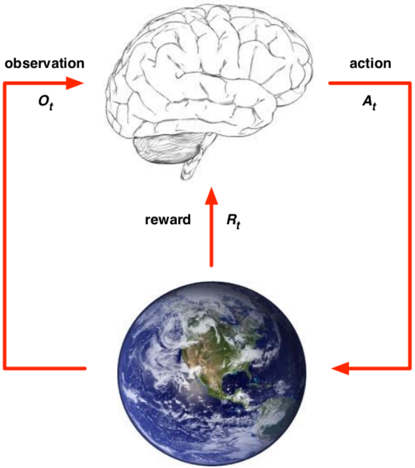
- Reward, R_t, is a **scalar** feedback signal
  - Signals how well agent is doing at time t
  - Agent maximizes the long run sum of rewards
  - Exogenously given

- Reward Hypothesis
  - All goals can be described by the maximisation of expected cumulative reward

Reward hypothesis

- Reward, R_t, is a **scalar** feedback signal
  - Signals how well agent is doing at time t
  - Agent maximizes the long run sum of rewards
  - Exogenously given
- Reward Hypothesis
  - All goals can be described by the maximisation of expected cumulative reward
- Examples
  - Pain if you lose a body part, satisfaction from food
  - Negative reward for moving in the gridworlds
  - Positive/negative reward for increasing/decreasing score in Atari videogames

Agent and environment



Source: David Silver lectures

History and State

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it
    - E.g. this might be a true cost function of hyperparameters

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it
    - E.g. this might be a true cost function of hyperparameters
  - The agent state S_t^a, internal representation

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it
    - E.g. this might be a true cost function of hyperparameters
  - The agent state S_t^a, internal representation
    - Important part, used by algorithms

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it
    - E.g. this might be a true cost function of hyperparameters
  - The agent state S_t^a, internal representation
    - Important part, used by algorithms
    - E.g. agent might use hyperparameter values to estimate the cost function

History and State

- **The history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards

- **The state** is a summary information of the history, some function of it

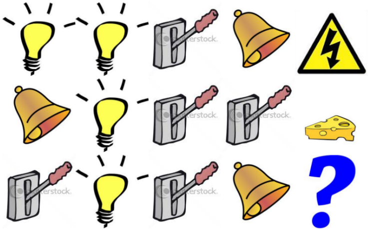
$$S_t = f(H_t)$$

  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it
    - E.g. this might be a true cost function of hyperparameters
  - The agent state S_t^a, internal representation
    - Important part, used by algorithms
    - E.g. agent might use hyperparameter values to estimate the cost function
    - Many choices, what to remember and what to throw away of H_t

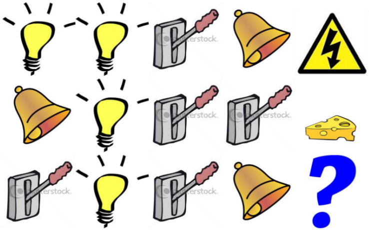
History and State

- **The history** is the sequence of observations, actions, rewards
$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$
  - The agent selects action A_t based on H_t
  - The environment selects observations and rewards
- **The state** is a summary information of the history, some function of it
$$S_t = f(H_t)$$
  - The environment state S_t^e, private representation of H_t
    - Agents might or might not observe parts of it
    - E.g. this might be a true cost function of hyperparameters
  - The agent state S_t^a, internal representation
    - Important part, used by algorithms
    - E.g. agent might use hyperparameter values to estimate the cost function
    - Many choices, what to remember and what to throw away of H_t
    - E.g. estimate function in parametric way and keep parameters

What is the agent's state?

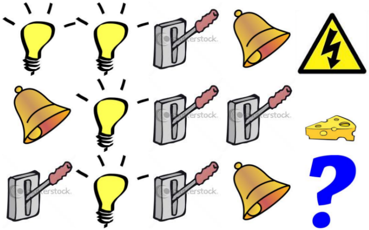


What is the agent's state?



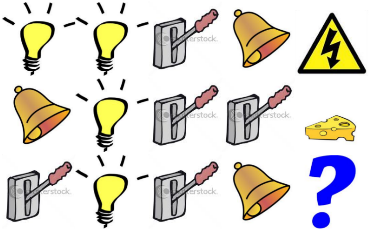
- Last 3 items in sequence?

What is the agent's state?



- Last 3 items in sequence?
- Counts for lights, bells and levers?

What is the agent's state?



- Last 3 items in sequence?
- Counts for lights, bells and levers?
- Complete sequence?

More about environments

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

- Partially observable environment (POMDP)

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

- Partially observable environment (POMDP)
  - Agent can indirectly observe environment state

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

- Partially observable environment (POMDP)
  - Agent can indirectly observe environment state
  - Using this info agent constructs the state

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

- Partially observable environment (POMDP)
  - Agent can indirectly observe environment state
  - Using this info agent constructs the state
    - E.g. beliefs of environment state: $S_t^a = (P[S_t^e = s^1], ..., P[S_t^e = s^n])$

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

- Partially observable environment (POMDP)
  - Agent can indirectly observe environment state
  - Using this info agent constructs the state
    - E.g. beliefs of environment state: $S_t^a = (P[S_t^e = s^1], ..., P[S_t^e = s^n])$
  - E.g. in hyperparameter case, we partially observe environment state through hyperparameter values

More about environments

- A state S_t is Markov if and only if

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t]$$

  - The future is independent of the past given the present
  - We have all the information necessary for making optimal choices

- Fully observable environment
  - Agent can observe environment state $O_t = S_t^a = S_t^e$
  - This is a Markov decision process (MDP)

- Partially observable environment (POMDP)
  - Agent can indirectly observe environment state
  - Using this info agent constructs the state
    - E.g. beliefs of environment state: $S_t^a = (P[S_t^e = s^1], ..., P[S_t^e = s^n])$
  - E.g. in hyperparameter case, we partially observe environment state through hyperparameter values
  - E.g. investment agent observes prices, but not trends etc

Constructing the Agent

Constructing the Agent

- **Policy**:
  - Agent's behaviour function
  - Deterministic policy: $a = \pi(s)$
  - Stochastic policy: $\pi(a|s) = P[A_t = a|S_t = s]$

Constructing the Agent

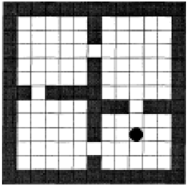
- **Policy**:
  - Agent's behaviour function
  - Deterministic policy: $a = \pi(s)$
  - Stochastic policy: $\pi(a|s) = P[A_t = a|S_t = s]$

- **Value function**:
  - Agent uses it to predict future reward, determines how good is each state and/or action
  - Used to select between actions
  - $V_\pi(s) = E_\pi[R_{t+1} + \gamma R_{t+2} + ...|S_t = s]$

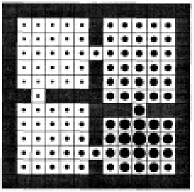
Constructing the Agent

- **Policy**:
  - Agent's behaviour function
  - Deterministic policy: $a = \pi(s)$
  - Stochastic policy: $\pi(a|s) = P[A_t = a|S_t = s]$
- **Value function**:
  - Agent uses it to predict future reward, determines how good is each state and/or action
  - Used to select between actions
  - $V_\pi(s) = E_\pi[R_{t+1} + \gamma R_{t+2} + ...|S_t = s]$
- **Model**: agent's representation of the environment, predicts
  - What the environment will do next
  - The next state: $\mathcal{P}^a_{ss'} = P[S_{t+1} = s'|S_t = s, A_t = a]$
  - The next reward: $\mathcal{R}^a_s = E[R_{t+1}|S_t = s, A_t = a]$

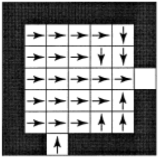
Gridworld example



Initial values



Iteration #5



Source: Sutton, Precup & Singh (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence, 112 (1-2), 181-211.

Types of agents

- Value Based: No Policy, Value Function
- Policy Based: Policy, No Value Function
- Actor Critic: Policy, Value Function

Types of agents

- Value Based: No Policy, Value Function
- Policy Based: Policy, No Value Function
- Actor Critic: Policy, Value Function

- Model-Free: Policy and/or Value Function, but no Model
- Model-Based: Policy and/or Value Function, Model

Exploration exploitation problem

- Acting involves a fundamental trade-off:
  - **Exploitation**: Make the best decision given current information
  - **Exploration**: Gather more information

Exploration exploitation problem

- Acting involves a fundamental trade-off:
  - **Exploitation**: Make the best decision given current information
  - **Exploration**: Gather more information
- The best long-term strategy may involve short-term sacrifices

Exploration exploitation problem

- Acting involves a fundamental trade-off:
  - **Exploitation**: Make the best decision given current information
  - **Exploration**: Gather more information
- The best long-term strategy may involve short-term sacrifices
- **Goal**: Gather enough information to make the best overall decisions

Exploration exploitation problem

- Acting involves a fundamental trade-off:
  - **Exploitation**: Make the best decision given current information
  - **Exploration**: Gather more information
- The best long-term strategy may involve short-term sacrifices
- **Goal**: Gather enough information to make the best overall decisions
- Examples:
  - Going to a favourite restaurant (**exploitation**), or try a new restaurant (**exploration**)
  - Show the most successful ad (**exploitation**), or show a new ad (**exploration**)

How can we try to solve it?

How can we try to solve it?

1. **Random exploration**
   - Adding some noise to a greedy policy
   - Examples: ϵ-greedy, Softmax

How can we try to solve it?

1. **Random exploration**
   - Adding some noise to a greedy policy
   - Examples: ϵ-greedy, Softmax

2. **Optimism in the face of uncertainty**
   - Using all available information, estimate uncertainty on value
   - Prefer to explore uncertain states/actions
   - Examples: Optimistic initialisation, Upper Confidence Bound, Thompson sampling, Expected Improvement, Probability of Improvement

How can we try to solve it?

1. **Random exploration**
   - Adding some noise to a greedy policy
   - Examples: ϵ-greedy, Softmax

2. **Optimism in the face of uncertainty**
   - Using all available information, estimate uncertainty on value
   - Prefer to explore uncertain states/actions
   - Examples: Optimistic initialisation, Upper Confidence Bound, Thompson sampling, Expected Improvement, Probability of Improvement

3. **Information state space search**
   - Considering agent's information in its state space
   - Lookahead to determine how information helps in maximizing rewards
   - Examples: Gittins indices (see Whittle, 1980), tractable approximation with Bayesian Adaptive Monte Carlo Planning (Guez, Silver, Dayan, 2012; 2014)