DAT440 / DIT470 / DIT471 Advanced topics in machine learning - Re-exam

Course Responsible and Examiner: Morteza Haghir Chehreghani

August 15, 2022, 2:00 pm

- You explain your solutions, for any question that the calculations are needed you must show the steps.
- You may have a cheat sheet of two pages (either one double-sided sheet or two one-sided sheets) in A4 format (font size 11 if typed, and similar font size if it is handwritten). You can include whatever you want in the cheat sheet. Besides that, you may not use any other resource or reference to answer the questions.
- You may have a simple standard calculator commonly used at Chalmers, though the questions do not need use of a calculator.
- Read the questions carefully such that you do not miss any question and ensure you clearly give the answer required for each (sub)question.
- You do not need to write (Python) code for any question.
- The exam grade and the final grade will be computed according to the formula mentioned on the course webpage. Accordingly, your grade will follow the distribution: 28 out of 60 (3,G), 36 out of 60 (4), 48 out of 60 (5, VG).
- There will be visits of the exam premises on two occasions to answer the clarification questions: one hour after the start of the exam and when an hour of the exam remains.

1. (10 points) Multiple choice questions: For each question you have to select **all options that are true for full score (and none if all are false)**. Fill in your answers in the form on the final page. **Make sure to include it with your exam.** The points do not necessarily reflect the number of correct answers.

   For the RL-related questions, consider an arbitrary action A_t taken according to policy π at state S_t which results in reward R_{t+1} and new state S_{t+1}. Then, $v_\pi(S_t)$ indicates the state values of state S_t under π and γ is the respective discount factor as introduced in the course.

   (a) (2 points) Let $v_k(s)$ be an approximation of the state value at state s. What paradigm best explains the following update?

   $$v_{k+1}(s) = \max_a \mathbb{E}\left[R_{t+1} + \gamma v_k(S_{t+1}) | S_t = s, A_t = a\right]$$

   i. Monte Carlo
   ii. Dynamic Programming
   iii. Temporal Difference

   (b) (3 points) What holds for the return $G_t = \sum_{k=t+1}^{T} \gamma^{k-1} R_k$?
   i. It is an unbiased estimate of the expected return $v_\pi(S_t)$

ii. It is a biased estimate of the expected return $v_\pi(S_t)$
iii. It has lower variance than the TD target
iv. It has higher variance than the TD target

(c) (1 point) The Markov property means that the future state depends on all previous states.
i. True
ii. False

(d) (3 points) Which of the following statements is/are correct about bandits and conjugate priors?
i. Conjugate priors are used in both Thompson Sampling and UCB to sample rewards.
ii. Conjugate priors make the computations more convenient.
iii. With conjugate priors, one does not need to do sampling in Thompson Sampling.

(e) (1 point) The UCB algorithm for the K-armed bandit problem *adapts* the exploration to the observed rewards/actions.
i. True
ii. False

2. (11 points) Successive Elimination

Consider a stochastic K-armed bandit problem with a set of arms $\mathcal{A}$, and $|\mathcal{A}| = K$. We define, at time t, the arm $a_t \in \mathcal{A}$ played by the agent, the reward $x_t(a) \in \{0, 1\}$ received by an agent if it plays arm $a \in \mathcal{A}$, the cumulative reward $s_t(a) = \sum_{j=1}^{t} \mathbb{1}\{a_j = a\} x_t(a)$ of arm a until time t, the number of times $n_t(a) = \sum_{j=1}^{t} \mathbb{1}\{a_j = a\}$ that arm a has been played until time t, the average reward $\hat{\mu}_t(a) = s_t(a)/n_t(a)$ of arm a until time t. We also define, for each arm $a \in \mathcal{A}$:

$$\mathrm{UCB}_t(a) = \hat{\mu}_{t-1}(a) + \sqrt{\frac{2 \ln T}{n_{t-1}(a)}}$$

$$\mathrm{LCB}_t(a) = \hat{\mu}_{t-1}(a) - \sqrt{\frac{2 \ln T}{n_{t-1}(a)}}$$

Algorithm 1 describes the Successive Elimination method, where each "phase" of the algorithm comprise lines $3 - 4$.

Algorithm 1 Successive Elimination

1: Initially all arms $a \in \mathcal{A}$ are set "active".
2: **while** time $t \leq T$ **do**
3: Play each remaining "active" arm (note that the time t is incremented *after each* arm played).
4: Deactivate each arm $a \in \mathcal{A}$ where there exists $a' \in \mathcal{A}$ such that $\mathrm{UCB}_t(a) < \mathrm{LCB}_t(a')$
5: **end while**

Consider the scenario in Table 1, where an agent has acted using Successive Elimination in a multi-armed bandit environment until time $t = 897$ (with horizon $T = 1100$). The 4 arms in the table are the remaining "active" arms at the beginning of the current "phase".

Table 1: Multi-armed bandit scenario

a	1	2	3	4
$s_{t-1}(a)$	112	56	168	84
$n_{t-1}(a)$	224	224	224	224

(a) (2 points) Which arms will be played by Successive Elimination during the current "phase" (i.e., beginning with time $t = 897$)? Briefly explain why.

(b) (3 points) Calculate $\text{UCB}_t(a)$ and $\text{LCB}_t(a)$ for each of the arms in Table 1 (note that $\ln T \approx 7$).

(c) (4 points) For each arm $a \in \mathcal{A}$, let $\tau_a \geq t$ be the first time step in which arm a is played after (and including) time step t. Then, let $x_{\tau_1} = 1$, $x_{\tau_2} = 0$, $x_{\tau_3} = 1$, and $x_{\tau_4} = 0$ (in other words, if arm 1 is played during the current "phase", it will receive a reward of 1, etc.). Which arms will remain "active" after the current "phase"? Show why.

(d) (2 points) Briefly explain why it can be reasonable to permanently deactivate some arms in early "phases".

3. (8 points) Thompson Sampling

(a) (2 points) Briefly explain how the Thompson Sampling algorithm balances exploration and exploitation.

(b) (2 points) Consider Thompson Sampling being applied to a problem with three arms, where the respective posterior distributions are Gaussian with variances equal to 1. The current estimates of the means are 0.4, 0.8 and 0.5. Which arm will be more likely to be selected the next time? Is that choice guaranteed to occur? Briefly explain your answer.

(c) (2 points) Under which conditions does Thompson Sampling perform similar to the greedy method? Briefly explain your answer.

(d) (2 points) Consider a variant of Thompson Sampling where, from each arm, two i.i.d. samples are drawn (instead of one), but only the second sample is used. What can you say about the performance of this variant compared to the standard Thompson Sampling method in terms of expected (average) regret? Briefly explain your answer.

4. (4 points) Policy improvement

Consider a deterministic policy $\pi(s)$. Show that if a new policy $\pi'(s)$ is greedy with respect to $v_\pi(s)$, then it must hold that $v_{\pi'}(s) \geq v_\pi(s)$ for all states s, with equality only if π' is an optimal policy. Explain every step in your proof.

5. (14 points) Deep Q-network

Algorithm 2 shows the Deep Q-network (DQN) algorithm. Answer the following questions about this algorithm.

(a) (2 points) Is it on-policy or off-policy? Briefly explain your answer.

(b) (3 points) What part(s) of the algorithm is/are related to learning the state-action value functions? Explain your answer.

(c) (7 points) Which part(s) of the algorithm suggests using the same data possibly multiple times? Briefly explain your answer. In addition, mention and explain three advantages of using this mechanism.

(d) (2 points) Explain why *duplicate network* is used.

6. (13 points) REINFORCE

Algorithm 3 shows the REINFORCE algorithm. Answer the following questions about this algorithm.

(a) (2 points) Is it on-policy or off-policy? Briefly explain your answer.

(b) (4 points) What are the similarities and dissimilarities with the (one-step) Actor-Critic algorithm? Briefly explain your answer.

(c) (2 points) Is it a Monte-Carlo, Temporal Difference learning or Dynamic Programming algorithm? Briefly explain your answer.

(d) (5 points) Assume the non-discounted case ($\gamma = 1$). Use the policy gradient theorem (Equation (1)) to derive the update step of $\boldsymbol{\theta}$. Explain every step in your derivation.

$$\nabla J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi(a|s,\boldsymbol{\theta}) \tag{1}$$

Algorithm 2 Deep Q-network (DQN)

Inputs:
  Number of updates to fix weights of target network C
  Discount factor γ
  Number of transitions in minibatch N
Initialize:
  Network $\hat{q}$
  Target network $\tilde{q}$
  Experience replay memory D
while not converged **do**
  $\epsilon \leftarrow$ setting new epsilon with ϵ-decay
  Choose action a from state s using policy ϵ-greedy($\hat{q}$)
  Take action a, observe reward r and next state s'
  Store transition $(s, a, r, s', done)$ in the experience replay memory D
  if enough experiences in D **then**
    Sample a random minibatch of N transitions from D
    for every transition $(s_i, a_i, r_i, s'_i, done_i)$ in minibatch **do**
      if $done_i$ **then**
        $y_i = r_i$
      else
        $y_i = r_i + \gamma \max_{a' \in \mathcal{A}} \tilde{q}(s'_i, a')$
      end if
    end for
    Calculate the loss $\mathcal{L} = \frac{1}{N} \sum_{i=0}^{N-1} \left(\hat{q}(s_i, a_i) - y_i\right)^2$
    Update $\hat{q}$ using the SGD algorithm by minimizing the loss $\mathcal{L}$
    Every C steps, copy weights from $\hat{q}$ to $\tilde{q}$
  end if
end while

Algorithm 3 REINFORCE

Inputs:
  Differentiable policy parameterization $\pi(a|s,\boldsymbol{\theta})$
  Discount factor γ
  Step size $\alpha > 0$
Initialize:
  Policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$
while not converged **do**
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$ by following $\pi(\cdot|\cdot,\boldsymbol{\theta})$
  for each step of episode $t = 0, 1, \ldots, T-1$ **do**
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t,\boldsymbol{\theta})$
  end for
end while