REMIDY

Nama : Alief Riskiawan Nama Samon
Nim : 192382005
Id : 31
Kelas : Alan Turing
Domain : Reinforcement Learning

1. Ringkasan Materi RL dari pertemuan pertama sampai dengan pertemuan ke 6
   a. Materi pertama (Introduction to RL)

      Definisi dan sejarah RL dimulai experimen yang dilakukan oleh Walter Mischel, seorang profesor di Universitas Stanford mengenai Stanford Marshmallow adalah sebuah studi tentang gratifikasi dan sempat tertunda karena pada tahun 1972. Dalam penelitian ini, seorang anak ditawari pilihan antara satu hadiah kecil namun langsung, atau dua hadiah kecil jika mereka menunggu untuk jangka waktu tertentu. Selama waktu ini, peneliti meninggalkan ruangan selama sekitar 15 menit dan kemudian kembali. Hadiahnya adalah marshmallow atau tongkat pretzel, tergantung pada preferensi anak. Dalam studi lanjutan, para peneliti menemukan bahwa anak-anak yang mampu menunggu lebih lama untuk hadiah yang disukai cenderung memiliki hasil hidup yang lebih baik, sebagaimana diukur dengan skor SAT, pencapaian pendidikan, indeks massa tubuh (BMI), dan ukuran kehidupan lainnya. Upaya replikasi dengan sampel dari populasi yang lebih beragam, lebih dari 10 kali lebih besar dari penelitian asli, hanya menunjukkan setengah dari efek dari penelitian asli. Replikasi menunjukkan bahwa latar belakang ekonomi, bukan kemauan keras, menjelaskan separuh lainnya. Kekuatan prediktif dari tes marshmallow ditantang dalam sebuah studi tahun 2020.

      Adapun algoritma dari RL yaitu agent berinteraksi terhadap environment dengan melakukan sebuah tindakan dari satu kondisi ke kondisi yang lain. Agen akan menerima reward berdasarkan aksinya, berdasarkan reward tersebut, agen akan mengetahui apakah tindakan yang dilakukannya itu baik atau tidak baik. Jika aksinya baik maka akan menerima reward positif dan jika aksinya tidak baik maka akan menerima reward negatif (lose). Contoh algoritma RL adalah objective (memaksimalkan skor permainan), state (menunjukan gambar dengan resolusi tertentu), action (memetakan pengendalian aksi daripada game tersebut), reward (skor permaianan), terminator (kalah dari permainan).

      Elemen enviroment pada RL dibagi menjadi 4 sub elemen sebagai pnyusun utam sistem RL yaitu, Sub elemen utama : a). Policy, b). Reward signal, c). Value function, d). Model environment. Aplikasi RL yaitu seperti AI of game, robotics, dan optimization

b. Pertemuan kedua (Markov Decision Proses dan Dynamic Programming)

Markov decision proses merupakan sebuah tuple yaitu S,A,P,R,Y. Dimana S = state, A = action, P = probabillity, R = reward, dan Y = discount factor. MDP medeslripsikan secara formal lingkungan untuk RL, secara spesifik biasanya dibuat saat enviroment fully observable, hampir semua RL problems dapat di normalisasikan menggunakan MDP.

Aplikasi RL dalam Dunia Nyata

- Robotik: Memutuskan bergerak kemana.
- Alokasi Sumber Daya: Memutuskan apa yang diproduksi
- Pertanian : Menentukan apa yang ditanam, tapi tidak tahu cuaca dan hasil tanam

Markov Decision Process

• States: himpunan dari states
• $Sstart \in States$: state awal
• Actions(s): actions yang tersedia dari state s
• $P\ s, a, s\ '$) : probabilitas ke s' jika mengambil action a dari state s
• Reward (s, a, s') : idem dengan diatas
• isEnd(s) : apakah state terakhir
• $0 \leq \gamma \leq 1$ faktor diskon

Probabilitas (atau Transition)

- Transition Probabilitas $P\ s, a, s\ '$ menentukan kemungkinan dari state yang didatangi dalam s' jika melakukan action a dari state s

Solusi untuk MDP

• Policy adalah fungsi π yang memetakan untuk setiap state $s \in States$ ke action $a \in Actions(s)$

Evaluasi sebuah policy

• Mengikuti policy akan bisa menghasilkan jalur yang acak (mengapa?) • Return (utility) dari sebuah policy adalah Jumlah dari reward selama mengikuti jalur (nilai yang acak)

Return (Utility)

• Return adalah Gt total reward yang di-diskon dari setiap waktu t

Diskon Y

Apa guna diskon Y:

Biasanya Markov reward dan MDP mengandung diskon, mengapa?

• Secara matematik memberikan konstanta untuk diskon itu mudah
• Menghindari return tak terhingga dalam proses Markov yang berbentuk siklus
• Masa depan yang tidak pasti yang mungkin tidak diwakili model
• Dalam hal finansial, reward langsung lebih menarik disbanding reward nanti
• Perilaku binatang / manusia biasanya lebih memilih reward langsung
• Tetap memungkinkan untuk menggunakan Markov reward yang tidak didiskon (γ = 1) (jika semua alur tidak berulang)

c. Pertemuan ketiga (Monte Carlo Prediction)

Konsep monte carlo : tidak mengambil pengetahuan lebih lengkap dari environment, belajar daari experimen episode per episode baik itu eksperimen aktuan maupun simulasi, belajar dari episode-episode secara utuh dan independen, tidak bootstrapping, MC didefinisikan untuk jenis episodic enviroment, ide utama dari MC : value didapat dari rata-rata returns maka semakin banyak returns, nilai rata-ratanya diharapkan konvergen pada expected value, seluruh episode dipertimbangkan dalam MC, hanya satu pilihan setiap perpindahan state di MC, sedangkan DP mempertimbangkan smeua probabilitas transis pada setiap perpindahan state, estimasi untuk semua state adalah independent di MC, tidak bootstrap, waktu yang dibutuhkan untuk mengestimasi suatu state tidak bergantung pada jumlah total state.

Monte carlo estimation merupakan model yang tidak tersedia, perlu untuk mengestimasi action juga, selain hanya mengestimasi statenya, tidak seperti DP, pada mdel free algoritma, state saja tidak cukup untuk menentukan policy, pada MC, action diperlukandalam menetukan policy. Pada evaluation problem, policy evaluation problem mengestimasi q_π (s, a), melakukan action a, dan mengikiuti policy π. First-visit MC mengestimasi value dari pasngan state-action (s,a) dengan merata-rata returns saat pertama kali menemui (visit) state s dan mengambil action a dalam suatu periode. Every-visit MC mengestimasi value dari pasangan state-action (s,a) dengan merata-rata returns, setiap menemui (visit) state s dan mengambil action a dalam suatu episode, implikasinya, tidak semua state-action (s,a) akan ditemui (visited). Solusinya : maintaining exploration.

Maintaining exploration dilakukan dengan : episode-episode memulai dengan sebuah pasangan state-action (s,a). Mekanisme perrtama dari exploring starts. Setiap pasangan memiliki probabilitas yang tidak sama dengan nol untuk dipilih sebagai permulaan episode. Mekanisme kedua dari exploring statrs. Semua pasangan state-action (s,a) memiliki probabilitas tidak sama dengan nol untuk memilih semua action pada setiap state.

Monte carlo control yaitu, policy evaluation : episode-episode dikerjakan dengan mekanisme exploring starts. Laku, MC akan menghitung q_{π_k} untuk π_k. Policy environments : untuk setiap action-value function q, greedy policy-nya, untuk setiap s $\in$ S, memilih sebuah action yang merupakan action-value maksimal, yaitu π (s) = ar g max q (s,a). Policy improvement dilakukan dengan membentuk π_{k+1} sebagai greedy policy berdasarkan pada $q_{\pi k}$.

On-policy monte carlo adalah algoritma monte carlo yang melakukan evaluasi atau improvisasi dari policy yang digunakan untuk membuat keputusan. Dalam ε-greedy policy, hampir semua action yang dipilih, mempunyai action value estimasi yang masksimal, tetapi dengan probabilitas ε dalam memilih action, tidak dengan random/acak.

Off–policy MC adalah algoritme Monte Carlo yang melakukan evaluasi atau improvisasi dari policy yang berbeda dalam meng-generate data (behavior policy), sedangkan target policy-nya sama. Target policy adalah policy yang dipelajari oleh agent. Target policy ini adalah greedy policy yang patuh pada Q.

d. Pertemuan keempat (Temporal-Difference Learning)

TDL = Mengestimasi reward pada setiap Langkah (step) :

- ❑ Agent tidak memiliki pengetahuan tentang environment yang sedang di eksplorasi
- ❑ Agent belajar dari environment melalui metode trial-and-error
- ❑ TD Learning merupakan kombinasi antara Monte Carlo dan Dynamic Programming (DP)
- ❑ Monte Carlo tidak membutuhkan pemodelan terhadap environment
- ❑ Dynamic Programming (DP)
- ❑ Model Free Learning

Terminology :

- ❑ Gamma (γ): faktor diskon, nilanya memiliki rentang antara 0 dan 1. Semakin mendekati 1 atau semkin tinggi nilainya semakin sedikit perolehan diskon.
- ❑ Lambda (λ): variable kredit, nilainya antara 0 dan 1. Semakin mendekati 1 atau semakin tinggi nilainya semakin banyak kredit yang di peroleh
- ❑ Alpha (α): learning rate, nilainya antara 0 dan 1.
- ❑ Delta (δ): a change or difference in value.

Model Free RL Sumber gambar DOTA 2: Open AI Salah satu kelemahan model based RL adalah jumlah state dan action haruslah terbatas, dan mudah dijelaskan menggunakan transiton probability.

Value-based method

Generalized Policy Iteration (GPI) merupakan bagian penting dari reinforcement learning. Hanya saja kita harus tahu bagaimana cara memperbaharui atau mengupdate value function. Value-based methods: metode ini didasarkan pada temporal difference learning (TD, SARSA, Q-Learning), value function $V\ \pi\ atau\ V\ *$. $\pi\ V\ \boldsymbol{\pi}$ V greedy (V) evaluation improvement $\pi*$.

Dynamic Programming & Monte Carlo Method

Dynamic Programming

- ❖ Update per step, menggunakan bootstrapping
- ❖ Membutuhkan model environment
- ❖ Computation cost Monte Carlo Method
- ❖ Update per episode
- ❖ Model-free (tidak membutuhkan model environment)
- ❖ Sulit untuk di aplikasikan pada continuing task.

Temporal Difference Learning

Dynamic Programming

- ❖ Model-free (tidak membutuhkan model environment)
- ❖ Online learning (fully incremental method) Dapat diaplikasikan pada continuing task
- ❖ Better convergence time Lebih cepat mencapai konvergen dari pada MC

e. Pertemuan kelima (Q Learning dan Deep Q Learning)

Q-Learning – Off Policy TD Control

Q-Learning merupakan pengembangan RL yang menggunakan Q-values (disebut juga action-values) untuk meningkatkan kemampuan agent belajar agent berulang-ulang.

Konsep dasar Q-Learning: • Terinspirasi dari value iteration • Sample an action • Observe the reward and the next state • Take the action with the highest Q (Max Q)

Action dari setiap step dapat dihitung untuk menemukan action terbaik (best action). Untuk keperluan ini digunakan Q-Table.

Algoritma Q-Learning secara sederhana: 1. Tentukan current state = initial state. 2. Dari current state, cari dengan nilai Q terbesar. 3. Tentukan current state = next state. 4. Ulang Langkah (2) dan (3) hingga current state = goal state.

Q Learning

Objektif dari Q Learning adalah mencari policy yang optimal berdasarkan in the sense that the expected value of the total reward over all successive steps is the maximum achievable. Atau dengan kata lain, tujuan dari Q Learning adalah mencari policy yang optimal dan mencari optimal Q Values untuk setiap pasangan state-action.

Terminology

Policy (π) : Adalah sebuah fungsi yg memetakan setiap state ke action. Sehingga dengan mengikuti policy, agent bisa memilih action apa yg akan dia ambil pada state tertentu. Reward (r) : feedback atau umpan balik untuk agent. Reward dapat bernilai positif (berupa hadiah) atau negatif (berupa hukuman) dan juga nol (tidak ada tindakan apapun terhadap agent). Episodes: Ketika agent berakhir pada terminating state dan tidak bisa mengambil action apapun pada proses learning.

Q Function : adalah fungsi yang digunakan dalam Q-Learning untuk mencari Q-Value untuk setiap pasangan state dan action dan untuk menentukan apakah sebuah action akan baik jika dilakukan pada state tertentu.

Q-Table : adalah sebuah table yang digunakan untuk menyimpan Q-Value untuk setiap pasangan state dan action. Horizontal axis dari table ini merepresentasikan kumpulan action, dan vertical axis merepresentasikan kumpulan state yang ada. Jadi, dimensi dari table akan bergantung terhadap jumlah actions dan jumlah states.

Choose Action

Eksplorasi adalah suatu metode pemilihan action dimana agent akan melakukan pemilihan action secara random dengan tujuan ia bisa mengetahui informasi tentang environment secara mendalam. Sedangkan Eksploitasi adalah sebuah metode pemilihan action dengan memilih action yang mempunyai return (dalam hal ini Q-Value) paling besar.

f. Pertemuan keenam (Robotics)

Robotics adalah suatu disiplin ilmu yang mempelajari tentang konsep suatu robot. Robot merupakan mesin yang beroperasikan secara otomatis yang menggantikan usaha manusia, meskipun mungkin tidak menyerupai manusia dalam penampilan atau melakukan fungsi dengan cara yang mirip manusia.

Deep Q Learning

Pada Deep Q Learning kita akan menggantikan Q-Table menggunakan sebuah Neural Network yang biasa disebut dengan Deep Q Network atau DQN.

Deep Q Network

Deep Q Network adalah sebuah NN yang menerima states yg diberikan oleh environment sebagai input, lalu DQN akan menghasilkan output estimasi Q Values pada setiap actions yang dapat diambil pada state tersebut. Tujuan dari NN ini adalah untuk menghasilkan aproksimasi Q Function yg optimal. Goal dari NN ini adalah untuk minimize loss, lalu setelah menghitung loss bobot pada network akan diupdate menggunakan stochastic gradient descent dan backpropagation seperti neural network pada umumnya.

Experience Replay and Replay Memory

Dalam proses training DQN kita akan menggunakan sebuah teknik yg dinamakan dengan experience replay. Dengan experience replay, kita menyimpan experience dari agent untuk setiap time step ke dalam sebuah wadah yang bernama replay memory. Secara teori seluruh experience agent pada setiap time step akan disimpan pada replay memory. Sebenarnya dalam praktiknya, kita akan mendefinisikan besaran experience yang dapat ditampung oleh replay memory sebesar N, dan kita hanya akan menyimpan N terakhir experience dari agent.

2. Buatlah satu contoh aplikasi RL dalam bidang apapun dan ceritakan bagaimana aplikasi itu diterapkan (tanpa codingan)
   Aplikasi di bidang pertanian contohnya : irigasi tetes otomatis mengunakan remot control. Cara penerapannya : bisa dicontrol dari jarak jauh dengan menekan tombol hijau pada remot control yang sudah dibuat dengan pemasangan antena penghubung antara antena remot control dan antena yang ada pada katup pembuka air, sehingga ketika remot control ditekan katup air akan dengan sendirinya terbuka secara otomatis dan mengeluarkan air secara bertahap sesuai volume air yang diatur sebelumnya.
3. Buat tabel perbandingan dari metode dynamic Programing, metode Monte Carlo, SARSA, Q-Learning, dan Deep Q-Learning

Metode	Perbandingan
Dynamic Programming	Untuk mengoptimalkan proses pengambilan keputusan secara bertahap, perhitungan di tahap yang berbeda dihubungkan melalui perhitungan rekursif.
Monte Carlo	Untuk mensimulasikan berbagai perilaku sistem fisika dan matematika dan untuk mengevaluasi integral definit, terutama integral multidimensi dengan syarat dan batasan yang rumit.
SARSA	Perbedaan Temporal kebijakan yang sama π untuk memilih tindakan yang akan diambil untuk keadaan sekarang & masa depan.
Q-Learnig	Untuk mempelajari policy yang memberi tahu agen tindakan apa yang harus diambil dalam keadaan apa**.** Q-learning tidak memerlukan model dari environtment, dan dapat menangani masalah dengan transisi stokastik dan reward, tanpa memerlukan adaptasi.
Deep Q-Learning	Untuk memberikan rekomendasi video.

4. Buat dan cari salah satu contoh code dari aplikasi algoritma Dynamic Programming, Monte Carlo, SARSA, Q-Learning dan Deep Q-Learning, pilih salah satu yang paling mudah dan jelaskan cara kerja programnya

   Dynamic Programming

   ⇨ 0,1,1, 2, 3. Di sini, setiap angka adalah jumlah dari dua angka sebelumnya.
   Let n be the number of terms.
   1. If n <= 1, return 1.
   2. Else, return the sum of two preceding numbers.

   ⇨ Di sini, kita menghitung urutan fibonacci hingga istilah ke-5.
   1. The first term is 0.
   2. The second term is 1.
   3. The third term is sum of 0 (from step 1) and 1(from step 2), which is 1.
   4. The fourth term is the sum of the third term (from step 3) and second term (from step 2) i.e. 1 + 1 = 2.

5. The fifth term is the sum of the fourth term (from step 4) and third term (from step 3) i.e. 2 + 1 = 3.

⇨ Kita memiliki urutan 0,1,1, 2, 3. Di sini, kita telah menggunakan hasil dari langkah-langkah sebelumnya seperti yang ditunjukkan di bawah ini. Ini disebut pendekatan pemrograman dinamis.

F(0) = 0
F(1) = 1
F(2) = F(1) + F(0)
F(3) = F(2) + F(1)
F(4) = F(3) + F(2)

Cara Kerja Pemrograman Dinamis

Pemrograman dinamis bekerja dengan menyimpan hasil subproblem sehingga ketika solusinya diperlukan, karena sudah ada dan kita tidak perlu menghitung ulang. Teknik menyimpan nilai subproblem ini disebut memorisasi. Dengan menyimpan nilai dalam array, sehingga dapat menghemat waktu.

```
var m = map(0 → 0, 1 → 1)
function fib(n)
  if key n is not in map m
      m[n] = fib(n − 1) + fib(n − 2)
  return m[n]
```

Pemrograman dinamis dengan memorisasi adalah pendekatan top-down untuk pemrograman dinamis. Dengan membalikkan arah kerja algoritma yaitu dengan memulai dari kasus dasar dan bekerja menuju solusi, kita juga dapat menerapkan pemrograman dinamis secara bottom-up.

```
function fib(n)
  if n = 0
      return 0
  else
      var prevFib = 0, currFib = 1
      repeat n − 1 times
          var newFib = prevFib + currFib
          prevFib = currFib
          currFib  = newFib
  return currentFib
```