

Laporan Praktikum week 4

Nama : Muhamad Debi Priantoro

NIM : 224308014

Kelas : TKA-6A

Akun Github (Tautan) : <https://github.com/muhammaddebi12>

Student Lab Assistant : Muhammad Mahirul Faiq (214308043)

1. Judul Percobaan

Reinforcement Learning for Autonomous Control with DQN

2. Tujuan Percobaan

Tujuan dari praktikum ini yaitu :

1. Memahami Memahami konsep dasar Reinforcement Learning (RL) dalam sistem kendali.
2. Mengimplementasikan agen RL menggunakan algoritma *Deep Q-Network* (DQN).
3. Menggunakan *OpenAI Gym* sebagai simulasi lingkungan untuk pelatihan RL.
4. Melatih dan menguji agen RL untuk mengontrol lingkungan secara otonom.
5. Menggunakan GitHub untuk version control dan dokumentasi praktikum.

3. Landasan Teori

Reinforcement Learning (RL) adalah sebuah algoritma yang memungkinkan agen untuk mengumpulkan pengetahuan (dalam bentuk sinyal penguatan) guna memilih tindakan yang mengarah pada hasil terbaik yang diharapkan. Kemampuan *reinforcement learning* untuk secara mandiri menjelajahi lingkungan yang sangat dinamis dan stokastik dan mengembangkan, dengan mengumpulkan umpan balik evaluatif dari lingkungan, kebijakan kontrol yang optimal (Karimah, 2023). Tidak seperti *machine learning* yang membutuhkan dataset untuk proses training, pada *reinforcement learning* agen akan mengeksplorasi lingkungannya dan membuat keputusan berdasarkan nilai reward dan punishment yang diberikan ketika agen melakukan suatu aksi (Kurniawati dkk., 2021).

Setelah proses trial dan error, agen akan mempelajari aksi apa yang harus dilakukan untuk mendapatkan reward dengan nilai tertinggi. Sehingga agen akan memiliki kecenderungan untuk mengambil aksi tersebut jika berada pada suatu kondisi tertentu. *Epsilon greedy* merupakan salah satu algoritma yang digunakan untuk *learning process*. Algoritma ini menyeimbangkan proses eksploitasi dan eksplorasi berdasarkan nilai epsilon. Ketika agen melakukan eksplorasi lingkungan, agen akan mengambil aksi acak tanpa mepedulikan nilai reward atau punishment

yang didapatkan, sedangkan ketika agen berada dalam mode eksploitasi, maka agen akan melakukan aksi yang memberikan nilai reward tertinggi (Liu dkk., 2022). Epsilon dalam hal ini merupakan parameter yang menentukan apakah agen akan melakukan eksplorasi atau eksploitasi. Pada praktikum ini dengan mengimplemtasikan agen RL menggunakan algoritma *Deep Q-Network* (DQN). DQN adalah algoritma RL yang menggabungkan algoritma *Q-Learning* dengan *Deep Neural Network* (Putra dkk., 2024). *Q-Learning* dikategorikan dalam metode model-free yang tidak membutuhkan informasi yang lengkap tentang kondisi *environment*. Berbeda dengan metode *model-based*, *Q-Learning* memungkinkan agen untuk berada satu langkah ke depan. Agen berada dalam keadaan tertentu, mengambil tindakan tertentu, dan menerima value berdasarkan tindakan tersebut. Pada DQN, DNN digunakan untuk memodelkan nilai value tersebut dengan menggunakan *Neural Network*.

4. Analisis dan Diskusi

Analisis:

Pada **CartPole-v1**, tugas agen adalah menjaga keseimbangan tiang di atas gerobak/kotak selama mungkin. Lingkungan ini memberikan reward sebesar +1 untuk setiap langkah yang berhasil mempertahankan keseimbangan, sehingga proses pembelajaran dapat berlangsung lebih cepat. Dengan ***epsilon decay* yang relatif cepat (0.995)**, agen lebih cepat beralih dari **eksplorasi** ke **eksploitasi**, yang mempercepat konsistensi pengambilan keputusan berdasarkan pengalaman sebelumnya. ***Replay buffer* sebesar 2000** untuk menangani kompleksitas lingkungan ini, karena setiap episode memberikan banyak pengalaman yang dapat dipelajari dalam waktu singkat. Selain itu, batas maksimal episode 1000 membuat agen memiliki banyak kesempatan untuk belajar dan menyempurnakan kebijakan yang lebih optimal.

Sebaliknya, **MountainCar-v0** menghadirkan tantangan yang lebih besar karena agen harus menggerakkan mobil melewati bukit/gunung dengan cara mengayun bolak-balik untuk mendapatkan momentum yang cukup. Tantangan utama pada lingkungan ini adalah sistem *reward*-nya yang lebih menantang, di mana agen hanya mendapatkan ***reward* yang signifikan (+100) jika berhasil mencapai puncak**, sementara langkah-langkah lainnya menghasilkan ***reward* negatif** yang membuat agen lebih sulit memahami strategi yang benar. Oleh karena itu, parameter ***epsilon decay* diperkecil (0.999)** agar agen tetap mengeksplorasi lebih lama sebelum akhirnya mulai mengeksploitasi strategi yang sudah ditemukan. Selain itu, ***replay buffer* diperbesar menjadi 5000** untuk menyimpan lebih banyak pengalaman yang dapat membantu agen memahami pola pergerakan yang lebih kompleks. Dengan batas **maksimal 200 langkah per episode**, agen harus belajar lebih cepat untuk mencapai solusi optimal dalam ruang tindakan yang lebih terbatas dibandingkan dengan CartPole

Diskusi:

Perbedaan dalam hasil pembelajaran pada **CartPole-v1** dan **MountainCar-v0** menunjukkan bahwa setiap lingkungan memiliki karakteristik unik yang mempengaruhi strategi pelatihan agen. **CartPole-v1** lebih mudah dipelajari karena memiliki sistem *reward* yang langsung memberikan umpan balik positif untuk tindakan yang benar. Oleh karena itu, pendekatan eksplorasi yang lebih cepat dapat membantu agen belajar lebih efisien. Sebaliknya, **MountainCar-v0** memerlukan strategi yang lebih sabar karena *reward* hanya diperoleh secara signifikan saat mencapai tujuan. Selain itu, perbedaan dalam struktur *reward* mempengaruhi efektivitas teknik *Experience Replay* dan *Target Network Update*. Pada **CartPole**, *replay buffer* yang lebih kecil sudah cukup karena lingkungan memberikan banyak pengalaman positif dalam waktu singkat. Namun, pada **MountainCar**, *replay buffer* yang lebih besar diperlukan agar agen memiliki lebih banyak sampel untuk memahami strategi jangka panjang.

Dari hasil ini, dapat disimpulkan bahwa pendekatan *reinforcement learning* tidak bisa diterapkan secara seragam pada semua lingkungan. Setiap lingkungan memiliki karakteristik unik yang menuntut penyesuaian parameter agar agen dapat belajar secara optimal.

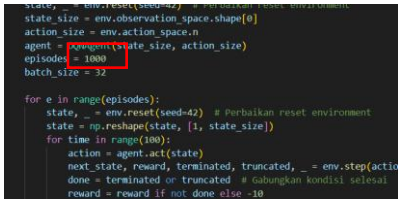
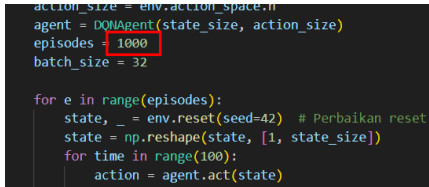
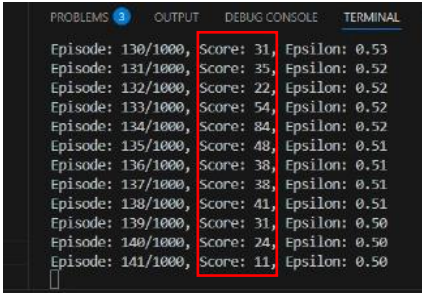
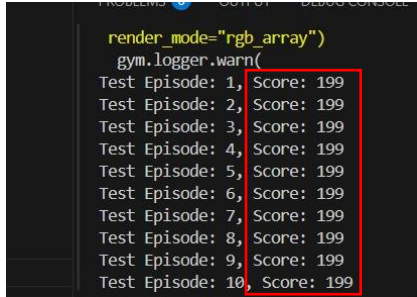
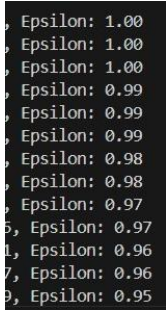
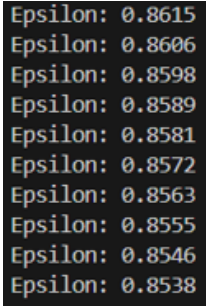
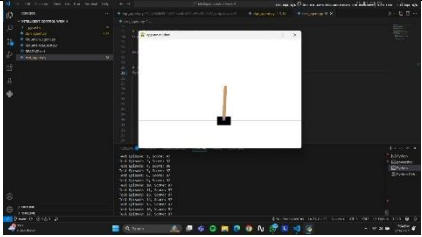
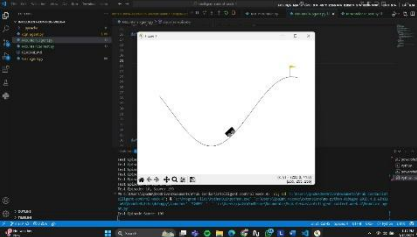
5. Assignment

Pada praktikum minggu keempat ini, telah dilakukan implementasi dan perbandingan performa agen *Deep Q-Network* (DQN) pada dua lingkungan berbeda, yaitu **CartPole-v1** dan **MountainCar-v0** menggunakan pustaka **OpenAI Gym** dan **TensorFlow**. Agen DQN yang digunakan memiliki **dua neural network** utama, yaitu **Q-Network** untuk menentukan aksi optimal berdasarkan kondisi lingkungan dan **Target Network** yang diperbarui secara berkala untuk stabilisasi pelatihan. Selain itu, agen juga menerapkan **Experience Replay** menggunakan *buffer* untuk menyimpan pengalaman sebelumnya dan meningkatkan efisiensi pembelajaran. Pada DQN standar, Q-network diupdate setiap kali menggunakan pengalaman yang dikumpulkan. Pembaruan ini sering kali tidak stabil karena pembaruan target (yang menggunakan Q-value yang diprediksi oleh jaringan yang sedang dipelajari) dapat menyebabkan fluktuasi yang besar, terutama jika jaringan telah mengalami pembaruan signifikan dalam langkah-langkah pelatihan sebelumnya. Target network digunakan untuk mengatasi masalah ini. Target network adalah jaringan Q yang terpisah dan diperbarui secara periodik, bukan setelah setiap langkah pelatihan. Jaringan ini digunakan untuk menghitung target Q-value dalam pembaruan DQN. Dengan cara ini, target Q-value lebih stabil karena tidak terus-menerus berubah seiring pembaruan jaringan Q yang sedang dilatih. Pembaruan target network dilakukan setiap sejumlah langkah tertentu (misalnya, setiap 1000 iterasi). Ini mengurangi fluktuasi yang terjadi pada pembaruan parameter jaringan, sehingga pelatihan menjadi lebih stabil. Pada eksperimen pertama dengan **CartPole-v1**, agen bertugas menjaga keseimbangan tiang di atas gerobak/kotak selama mungkin. Agen mendapatkan *reward* +1 setiap langkah, sehingga umpan balik yang sering memungkinkan pembelajaran lebih cepat. Parameter seperti **epsilon decay (0.995)**, **replay buffer (2000)**, dan **batch size (64)** telah disesuaikan agar eksplorasi berkurang lebih cepat dan agen dapat fokus mengeksplorasi kebijakan yang sudah dipelajari. Hasil pelatihan menunjukkan bahwa agen dapat meningkatkan skornya secara bertahap dan belajar menjaga keseimbangan tiang lebih lama seiring berjalannya waktu.

Sedangkan, eksperimen kedua dilakukan pada **MountainCar-v0**. **MountainCar-v0** agen harus menggerakkan mobil di lereng gunung untuk mencapai titik tertinggi. Namun, masalah utama adalah mobil awalnya tidak cukup kuat untuk langsung mencapai puncak, sehingga agen harus belajar untuk memanfaatkan gaya gravitasi untuk memperoleh kecepatan yang cukup. Ini adalah environment dengan *reward* yang lebih jarang, yang dapat menguji seberapa baik agen dapat mengeksplorasi lingkungan dan mengoptimalkan kebijakan eksplorasi, yang memiliki tantangan lebih besar karena agen harus menggerakkan mobil untuk mencapai puncak bukit dengan memanfaatkan momentum. Tidak seperti *CartPole*, *reward* hanya diberikan jika agen berhasil mencapai puncak, sehingga strategi eksplorasi menjadi lebih sulit. Oleh karena itu, beberapa parameter diubah, seperti **epsilon decay yang lebih lambat (0.999)**, **replay buffer yang lebih besar (5000)**, dan **tambahan reward +100 ketika berhasil mencapai tujuan**. Hal ini bertujuan untuk memberi agen lebih banyak kesempatan untuk menemukan strategi optimal sebelum memasuki fase eksploitasi. Dengan perubahan ini, agen dapat secara bertahap belajar menggunakan momentum untuk mencapai puncak meskipun membutuhkan waktu lebih lama dibandingkan dengan *CartPole*. *MountainCar-v0*, meskipun *environment* ini lebih sederhana, agen yang menggunakan target network juga menunjukkan kinerja yang lebih stabil dan konsisten, terutama dalam mengatasi tantangan eksplorasi yang lebih sulit dihadapi oleh agen tanpa target network. Perbandingan ini memperlihatkan bahwa penggunaan target network pada DQN dapat mengurangi ketidakstabilan yang sering terjadi selama pelatihan, memungkinkan agen untuk lebih cepat beradaptasi dan mencapai performa yang lebih optimal dalam berbagai jenis *environment*. Secara keseluruhan, tugas ini menunjukkan bagaimana **pemilihan parameter dalam reinforcement learning sangat bergantung pada karakteristik lingkungan**. Lingkungan dengan *reward* yang lebih sering, seperti *CartPole*, memungkinkan agen belajar lebih cepat dengan **epsilon decay yang lebih cepat dan replay buffer yang lebih kecil**. Sebaliknya, lingkungan dengan *reward* yang jarang seperti *MountainCar*, membutuhkan **eksplorasi yang lebih lama dan replay buffer yang lebih besar** agar agen dapat mengumpulkan pengalaman yang cukup untuk menemukan strategi yang efektif.

6. Data dan Output Hasil Pengamatan

Hasil Pengamatan :

| No. | Variabel | Hasil Pengamatan | |
|-----|-------------------------|---|---|
| | | CartPole-v1 | MountainCar-v0 |
| 1. | Waktu Pelatihan | CartPole lebih cepat mencapai performa optimal dalam beberapa ratus episode | MountainCar tetap membutuhkan lebih banyak waktu untuk mencapai keberhasilan yang stabil |
| 2. | Jumlah Episode Maksimum | CartPole dilatih hingga 1000 episode  | MountainCar dilatih hingga 1000 episode  |
| 3. | Skor Awal dan Maksimum | Skor bervariasi tergantung seberapa lama tiang tetap seimbang  | Skor awal mencapai 199 karena maksimal langkah dalam satu episode adalah 200  |
| 4. | Epsilon | Nilai epsilon (ε) menurun seiring dengan proses agen  | Nilai epsilon (ε) menurun seiring dengan proses agen  |
| 5. | Kinerja Agen | Kinerja yang lebih stabil dengan skor yang lebih tinggi | Masih menghadapi tantangan dalam mencapai puncak |
| 6. | Kondisi Berhasil | Episode berakhir jika tiang jatuh atau batas langkah maksimum tercapai | Episode berakhir jika mobil mencapai puncak bukit (flag) |
| 7. | Hasil |  |  |

7. Kesimpulan

Dari percobaan yang telah dilakukan dapat disimpulkan bahwa:

- CartPole-v1 lebih cepat dipelajari karena memberikan *reward* di setiap langkah, sehingga agen dapat dengan cepat memahami strategi optimal.
- MountainCar-v0 lebih sulit dipelajari karena *reward* hanya diberikan saat mencapai puncak bukit, memerlukan strategi eksplorasi lebih lama, dan mengandalkan momentum untuk berhasil.
- Penggunaan target *network* membantu stabilitas pembelajaran, menghindari perubahan nilai Q yang terlalu drastis.
- *Experience Replay* meningkatkan efisiensi pembelajaran, tetapi ukuran buffer harus disesuaikan dengan kompleksitas lingkungan agar tetap efektif.
- *Hyperparameter* sangat menentukan keberhasilan agen, sehingga perlu disesuaikan dengan karakteristik lingkungan untuk hasil optimal.

8. Saran

Untuk meningkatkan performa agen DQN pada lingkungan yang lebih kompleks, disarankan untuk menyesuaikan parameter lebih lanjut melalui eksperimen, seperti menggunakan *dueling DQN*, *prioritized experience replay*, atau peningkatan arsitektur jaringan dengan lebih banyak neuron atau lapisan. Selain itu, untuk kasus seperti MountainCar, pendekatan lain seperti *actor-critic methods* atau *policy-based reinforcement learning* dapat dicoba untuk meningkatkan efisiensi pembelajaran. Lebih jauh, penggunaan *hardware* yang lebih kuat seperti GPU dapat mempercepat proses pelatihan, terutama pada lingkungan dengan kompleksitas tinggi.

9. Daftar Pustaka

- Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Chacon, S., & Straub, B. (2014). *Pro Git* (Edisi ke-2nd).
- Mnih, V., Kavukcuoglu, K., Silver, D., dkk. (2015). Kontrol tingkat manusia melalui pembelajaran penguatan mendalam. *Nature*, 518(7540)
- Karimah Tauhid, Volume 2 Nomor 1 (2023), e-ISSN 2963-590X. (2023). 2.
- Kurniawati, N., Ningsih, Y. K., Puspa, S. D., & Adi, T. S. (2021). Algoritma Epsilon Greedy pada Reinforcement Learning untuk Modulasi Adaptif Komunikasi Vehicle to Infrastructure (V2I). *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, 9(3), 716. <https://doi.org/10.26760/elkomika.v9i3.716>
- Liu, F., Viano, L., & Cevher, V. (t.t.). Understanding Deep Neural Function Approximation in Reinforcement Learning via ϵ -Greedy Exploration.
- Putra, R. A., Syahbana, Y. A., & Ananda. (2024). Implementasi Algoritma Deep Q-Network (DQN) pada Lampu Lalu Lintas Adaptif Berdasarkan Waktu Tunggu dan Arus Kendaraan. *The Indonesian Journal of Computer Science*, 13(5). <https://doi.org/10.33022/ijcs.v13i5.4372>
-