

Laporan Praktikum 4 Kontrol Cerdas

Nama : Atniko Dwi Saputra

NIM : 224308005

Kelas : TKA – 6A

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

Student Lab Assistant : Muhammad Mahirul Faiq / 214308043

1. Judul Percobaan

Judul : Reinforcement Learning untuk Pengendalian Otonom dalam Simulasi

2. Tujuan Percobaan

Pada praktikum ini, mahasiswa akan dapat:

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

3. Landasan Teori

Reinforcement Learning (RL) adalah cabang dari machine learning yang berfokus pada bagaimana agen dapat belajar untuk mengambil keputusan optimal melalui interaksi dengan lingkungannya. Tidak seperti supervised learning yang menggunakan dataset berlabel, RL bekerja berdasarkan mekanisme trial-and-error, di mana agen mendapatkan reward atau punishment sesuai dengan aksi yang diambil. Model RL umumnya terdiri dari empat komponen utama: agen, lingkungan, aksi, dan fungsi reward.

Autonomous control mengacu pada sistem yang mampu mengambil keputusan dan beradaptasi tanpa intervensi manusia. Dalam konteks RL, autonomous control memungkinkan agen untuk mengendalikan sistem dinamis secara optimal dengan belajar dari pengalaman. Contohnya termasuk kendaraan otonom, robotika, dan pengendalian sistem energi.

RL sering dimodelkan menggunakan Markov Decision Process (MDP), yang terdiri dari:

- State (S): Kondisi sistem pada suatu waktu tertentu.
- Action (A): Aksi yang dapat dilakukan oleh agen.
- Reward (R): Umpan balik yang diterima setelah melakukan aksi.

- Transition Probability (P): Probabilitas berpindah dari satu state ke state lainnya setelah aksi tertentu.

Metode RL yang umum digunakan dalam autonomous control meliputi Q-Learning, Deep Q-Network (DQN), dan Policy Gradient Methods seperti Proximal Policy Optimization (PPO).

Meskipun RL menawarkan solusi adaptif, tantangan utamanya adalah eksplorasi dan eksploitasi, efisiensi sampel, serta kestabilan pelatihan. Seiring perkembangan teknologi, pendekatan seperti model-based RL, meta-learning, dan transfer learning semakin digunakan untuk meningkatkan efisiensi dan kemampuan adaptasi dalam sistem autonomous control.

Dengan kemajuan dalam komputasi dan algoritma, RL terus berkembang sebagai metode yang kuat untuk autonomous control di berbagai bidang, termasuk robotika, transportasi, dan sistem industri.

4. Analisis dan Diskusi

Analisis :

1. Bagaimana Performa Agen dalam Mengontrol Environment CartPole?

Performa agen dalam mengontrol environment CartPole bergantung pada seberapa baik agen dapat menyeimbangkan tiang di atas gerobak selama mungkin. Dengan model DQN (Deep Q-Network) yang telah dilatih, agen seharusnya dapat mencapai skor tinggi, mendekati 500 langkah per episode (batas maksimum). Jika agen masih dalam tahap pelatihan atau memiliki parameter suboptimal, performanya bisa lebih rendah, dengan episode berakhir lebih cepat karena kegagalan menjaga keseimbangan.

2. Bagaimana Perubahan Parameter Mempengaruhi Kinerja Agen?

Beberapa parameter penting dalam RL yang dapat mempengaruhi performa agen adalah:

- Gamma (γ - Discount Factor):
 - Gamma menentukan seberapa jauh agen menghargai reward di masa depan.
 - Gamma tinggi (misalnya 0.99) → Agen lebih fokus pada reward jangka panjang, sering menghasilkan strategi yang lebih optimal.
 - Gamma rendah (misalnya 0.8) → Agen lebih mementingkan reward langsung, bisa menyebabkan keputusan suboptimal.
- Epsilon (ϵ - Exploration Rate):
 - Epsilon mengontrol keseimbangan antara eksplorasi (mencoba aksi baru) dan eksploitasi (memanfaatkan aksi terbaik yang sudah diketahui).
 - Epsilon tinggi (misalnya 1.0 di awal pelatihan) → Agen lebih sering mengeksplorasi berbagai aksi, meningkatkan kemungkinan menemukan strategi optimal.
 - Epsilon rendah (misalnya 0.01 saat testing) → Agen lebih sering menggunakan kebijakan terbaik yang telah dipelajari, menghasilkan performa lebih stabil.
- Learning Rate (α):
 - Learning rate mengontrol seberapa besar perubahan pada DQN setiap kali agen belajar dari pengalaman.
 - Learning rate tinggi (misalnya 0.1) → Agen belajar lebih cepat, tetapi bisa menjadi tidak stabil.
 - Learning rate rendah (misalnya 0.001) → Agen belajar lebih lambat, tetapi hasilnya lebih stabil dalam jangka panjang.

3. Apa Tantangan yang Muncul Selama Pelatihan Agen RL?

Beberapa tantangan yang sering dihadapi selama pelatihan agen RL pada environment CartPole meliputi:

- Ketidakstabilan dalam Pembelajaran : Jika learning rate terlalu besar atau parameter tidak diatur dengan baik, agen bisa gagal mencapai konvergensi.

- Eksplorasi vs. Eksploitasi : Jika **ϵ -greedy policy** tidak dikonfigurasi dengan baik, agen bisa terjebak dalam strategi suboptimal karena eksploitasi terlalu dini atau eksplorasi yang tidak cukup.
- Overfitting ke Lingkungan Latihan : Agen yang terlalu banyak berlatih dalam satu environment mungkin kesulitan beradaptasi jika environment sedikit berubah.
- Efisiensi Sampel : **DQN** membutuhkan banyak episode untuk mencapai performa optimal, yang memakan banyak waktu dan sumber daya komputasi.

Diskusi :

1. Apa Perbedaan Utama antara Reinforcement Learning dan Metode Supervised Learning dalam Sistem Kendali?

- Supervised Learning
 - Membutuhkan dataset berlabel yang sudah tersedia sebelumnya.
 - Model belajar dengan memetakan input ke output berdasarkan contoh yang sudah diberikan.
 - Cocok untuk tugas yang bersifat prediksi, seperti klasifikasi atau regresi.
 - Contoh dalam sistem kendali: Prediksi suhu berdasarkan data sensor untuk sistem HVAC.
- Reinforcement Learning
 - Tidak memerlukan data berlabel, tetapi belajar melalui interaksi dengan lingkungan.
 - Agen mendapatkan reward sebagai umpan balik untuk menentukan strategi terbaik.
 - Cocok untuk pengambilan keputusan dalam lingkungan yang dinamis dan berubah-ubah.
 - Contoh dalam sistem kendali: Mengontrol robot agar berjalan stabil dengan belajar dari keseimbangan.

2. Bagaimana Strategi Eksplorasi (Exploration) dan Eksploitasi (Exploitation) Dapat Dioptimalkan pada Agen RL?

Beberapa cara optimasi strategi eksplorasi dan eksploitasi

1. Upper Confidence Bound (UCB) : memilih aksi berdasarkan keseimbangan antara eksplorasi dan ekspektasi reward.
2. Boltzmann Exploration : Menghindari eksplorasi yang sepenuhnya acak, sehingga lebih fokus pada aksi yang lebih menjanjikan.
3. Intrinsic Motivation : Agen diberi reward tambahan untuk mencoba sesuatu yang baru, mirip dengan bagaimana manusia belajar dari rasa ingin tahu.

3. Potensi Aplikasi RL dalam Sistem Kendali Nyata yang Dapat Diimplementasikan

Reinforcement Learning memiliki banyak aplikasi dalam sistem kendali di dunia nyata, termasuk:

1. Kendaraan Otonom
 - RL digunakan dalam self-driving cars untuk pengambilan keputusan seperti navigasi, penghindaran rintangan, dan manuver optimal.
2. Robotika Industri
 - Digunakan dalam pengendalian lengan robot untuk tugas seperti perakitan otomatis dan manipulasi objek dengan presisi tinggi.

5. Assignment

Pada praktikum ini, algoritma **Deep Q-Network (DQN)** digunakan untuk melatih agen dalam lingkungan **MountainCar-v0**. Tujuan utama agen adalah mencapai puncak bukit dengan memilih aksi yang paling optimal. Implementasi dimulai dengan mengimpor pustaka seperti **Gymnasium** untuk pengelolaan lingkungan, **NumPy** untuk operasi numerik, **random** untuk eksplorasi aksi,

serta **deque** dari **collections** untuk menyimpan pengalaman agen. Selain itu, **TensorFlow.Keras** digunakan dalam membangun dan melatih model jaringan saraf.

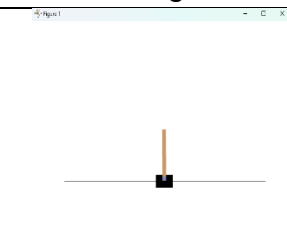
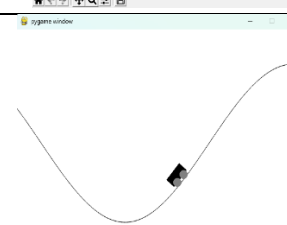
Agen DQN didefinisikan dalam kelas **DQNAgent**, yang diinisialisasi berdasarkan ukuran keadaan dan jumlah aksi yang tersedia. Beberapa parameter utama dalam algoritma ini meliputi **gamma** sebagai faktor diskon reward masa depan, **epsilon** untuk eksplorasi, serta **learning rate** yang menentukan kecepatan optimasi model.

Dalam eksperimen **CartPole-v1**, agen bertugas menjaga keseimbangan tiang di atas gerobak. Karena lingkungan ini lebih sederhana dan dapat diprediksi, agen bisa belajar dengan cepat. Setiap kali berhasil menjaga keseimbangan, agen mendapatkan reward positif. Pelatihan biasanya selesai dalam beberapa ratus episode karena agen relatif mudah mencapai performa optimal.

Sebaliknya, **MountainCar-v0** memiliki tantangan lebih kompleks. Agen harus membangun momentum dengan bergerak maju dan mundur sebelum dapat mencapai puncak bukit karena tenaga mobil tidak cukup untuk langsung sampai tujuan. Reward lebih sulit diperoleh, dan agen sering gagal di awal pembelajaran. Karena tingkat kesulitan yang lebih tinggi, **MountainCar-v0** memerlukan model yang lebih kompleks atau teknik tambahan untuk meningkatkan stabilitas pelatihan. Proses belajar dalam lingkungan ini membutuhkan lebih banyak episode, dengan nilai reward yang fluktuatif, sehingga diperlukan penyesuaian agar agen bisa mencapai performa optimal secara konsisten.

6. Data dan Output Hasil Pengamatan

Sajikan data dan hasil yang diperoleh selama percobaan. Gunakan tabel untuk menyajikan data jika diperlukan.

No	Variabel	Hasil Pengamatan
1	Cartpole.py	
2	Mountaincar.py	

7. Kesimpulan

CartPole-v1:

- Fokus pada stabilitas dan pengendalian keseimbangan tiang di atas gerobak.
- Reward diberikan setiap langkah yang sukses menjaga keseimbangan.
- Agen diuji dengan epsilon rendah untuk memastikan eksploitasi penuh dari model yang sudah dilatih.
- Menggunakan animasi untuk visualisasi kinerja agen.

MountainCar-v0:

- Tantangan lebih sulit karena reward standar hanya diberikan saat mobil mencapai puncak bukit.
- Algoritma diperbaiki dengan reward modifikasi untuk mendorong agen mencapai puncak lebih

cepat.

- Menggunakan replay memory dan decay pada epsilon untuk meningkatkan efisiensi eksplorasi.
- Training dilakukan dalam beberapa episode sebelum model digunakan untuk pengujian.

8. Saran

CartPole:

- Dapat ditingkatkan dengan menambahkan metode dueling DQN atau Double DQN untuk meningkatkan stabilitas pembelajaran.
- Bisa menggunakan frame skipping atau reward shaping untuk meningkatkan efisiensi.
- Mempertimbangkan implementasi Curiosity-driven Exploration untuk mengatasi eksplorasi suboptimal.

MountainCar:

- Menggunakan prioritized experience replay agar agen lebih cepat belajar dari pengalaman yang berpengaruh besar.
- Meningkatkan efisiensi dengan actor-critic methods (A2C atau DDPG) untuk menangani lingkungan dengan reward yang jarang.
- Menambahkan transfer learning dari model lain yang sudah dilatih di lingkungan serupa untuk mempercepat konvergensi.

9. Daftar Pustaka

- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction (2nd Edition)*. MIT Press.
- Silver, D. (2015). *Lecture Notes on Reinforcement Learning*. University College London (UCL).
- Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). *Human-level control through deep reinforcement learning*. *Nature*, 518(7540), 529–533.