Jurnal Rekayasa Sistem Informasi dan Teknologi
Volume 1, No 3 – Februari 2024
e-ISSN : 3025-888X

PENGAPLIKASIAN *DEEP REINFORCEMENT Q-LEARNING* UNTUK PREDIKSI PERDAGANGAN VALAS OTOMATIS

Abdillah Baradja[1], Tri Irianto Tjendrowasono[2]
[1,2]Fakultas Teknik Elektro dan Informatika, Universitas Surakarta, Indonesia
[1]dillahbaraja@gmail.com ; [2]tjendrowasono@gmail.com

Received: 10-02- 2024	Revised: 12-02-2024	Approved: 18-02-2024

ABSTRAK

B*ursa valas, pasar finansial terbesar di dunia dengan transaksi harian $5 triliun USD, adalah tempat investor dan pedagang membeli, menjual, dan menukar mata uang asing. Menghadapi tantangan membuat keputusan perdagangan yang tepat, inovasi teknologi seperti machine learning menawarkan solusi. Khususnya, Deep Reinforcement Learning (DRL), yang telah menunjukkan keunggulan atas manusia dalam berbagai tugas, termasuk game, menjanjikan potensi revolusioner dalam perdagangan valas. Menggunakan algoritma Deep Q-Network (DQN) Learning, penelitian ini bertujuan untuk mengembangkan model yang dapat memaksimalkan keuntungan dan meminimalkan risiko dalam lingkungan perdagangan yang kompleks dan dinamis. Kami menerapkan Deep Q-network (DQN) dan Deep Recurrent Q-network (DRQN) untuk mengembangkan sistem perdagangan harian otomatis pada pasangan mata uang EURUSD, memanfaatkan data perdagangan harian sebagai indikator lingkungan. Dalam penelitian kami, kinerja agen DRL dibandingkan dengan model tradisional dan DQN acak, menunjukkan bahwa algoritma DQN kami mengungguli model standar, sementara DRQN lebih superior, memanfaatkan pola tersembunyi dalam data urutan waktu. Hasil ini menekankan potensi penggunaan machine learning untuk menciptakan sistem perdagangan valas yang efisien dan menghasilkan keuntungan jangka panjang secara konsisten, menggambarkan langkah maju signifikan dalam teknologi perdagangan.*

Kata kunci: Machine Learning, Prediksi Harga Valas, Perdagangan Otomatis, Deep Reinforcement Learning

PENDAHULUAN

Pembelajaran mesin (*machine learning*) telah menjadi solusi utama dari sebagian besar masalah dalam perkembangan aplikasi teknologi informasi selama beberapa dekade terakhir [1]. Sebagian besar sistem pembelajaran mesin yang ada menggunakan pembelajaran terawasi (*supervised learning*) atau pembelajaran mendalam (*deep learning*) untuk memprediksi harga mata uang berdasarkan data pasar yang lampau. Sistem ini pada dasarnya membantu untuk membuat keputusan dan memberikan informasi perhitungan risiko kepada para pedagang mata uang asing [2].

Perkembangan dari *reinforcement learning* [3], [4] yang dapat mempelajari aturan dengan baik untuk masalah keputusan yang berurutan dan mengungguli manusia dalam banyak tugas seperti bermain game Atari [5], [6]. Reinforcement learning menggunakan bot sebagai agen yang belajar dari suatu lingkungan pada periode waktu tertentu. Agen tersebut akan menerima imbalan atau hukuman berdasarkan hasil keputusan yang diambil olehnya. Selanjutnya agen akan memutuskan tindakan berikutnya dengan tujuan memaksimalkan mendapatkan hadiah untuk setiap tindakan pada suatu lingkungan tersebut. Tujuan perdagangan mata uang asing adalah untuk menghasilkan keuntungan, maka percobaan yang kami lakukan tidak hanya memberikan informasi prediksi melainkan untuk membuat keputusan perdagangan tanpa campur tangan manusia dengan menggunakan *reinforcement learning* [7].

Kami mengeksplorasi algoritma *deep reinforcement learning* untuk menghasilkan model yang memaksimalkan keuntungan sekaligus meminimalkan resiko. dengan *Deep Q-Network* (DQN) *Learning* untuk menemukan pola yang menguntungkan di lingkungan perdagangan mata uang yang kompleks dan dinamis serta mempelajari pengaturan yang lebih baik untuk mencapai akumulasi keuntungan jangka panjang. Algoritma DQN terdiri dari komponen utama, (1) memodelkan fungsi nilai Q dari setiap pasangan keadaan dan tindakan, (2) meningkatkan stabilitas seluruh kerangka kerja selama pelatihan, (3) pengalaman pengulangan yang menghilangkan korelasi antara sample dan meningkatkan pemanfaatannya.

Keadaan lingkungan perdagangan yang terus bertambah dan terdapat informasi tersembunyi yang penting, membuat proses perdagangan lebih seperti *Proses Observable Markov Decision* (POMDP) [8].

Masalah penelitian yang akan diangkat dalam studi ini fokus pada pengembangan solusi untuk menghadapi tantangan dalam perdagangan algoritmik yang disebabkan oleh volatilitas tinggi dan noise pada data seri waktu keuangan. Masalah utama yang ditangani mencakup penciptaan metode penyaringan data untuk mengurangi noise dan meningkatkan kapabilitas generalisasi model perdagangan, pembuatan set aksi yang dinamis untuk mengadaptasi strategi perdagangan terhadap fluktuasi pasar, optimalisasi struktur penghargaan untuk memfokuskan pada keuntungan jangka panjang, serta penerapan teknik pengenalan pola temporal untuk memperkuat prediksi dan keputusan perdagangan berdasarkan analisis data historis dan tren pasar saat ini. Karenanya kami mengganti lapisan yang terhubung sepenuhnya di DQN dengan lapisan LSTM dengan ukuran yang sama, dan berhasil membuktikan bahwa pengenalan perulangan dapat meningkatkan kerja agen DQN di lingkungan perdagangan mata uang asing.

TINJAUAN PUSTAKA

Tipe penyelesaian masalah dengan menggunakan *Reinforcement learning* mencakup solusi untuk mempelajari apa yang harus dilakukan, bagaimana memetakan situasi menjadi aksi dengan tujuan memaksimalkan hadiah yang dihasilkan. Pada dasarnya adalah tindakan sistem pembelajaran untuk mendapatkan input nilai selanjutnya. Seperti dalam banyak bentuk pembelajaran mesin, sistem pembelajaran tidak mendapat informasi tentang tindakan yang direkomendasikan, melainkan harus menemukan sendiri tindakan mana yang menghasilkan hadiah paling banyak dengan mencoba semuanya. Untuk kasus yang lebih komplek, tindakan dapat mempengaruhi tidak hanya hadiah saja tetapi juga situasi berikutnya yang membuat *Reinforcement learning* penting dalam banyak aplikasi [9].

Dalam penelitian terbaru pada pembelajaran mesin yang saat ini dipelajari adalah *Supervised learning* yang melibatkan pembelajaran dari kumpulan data pelatihan berlabel. Contoh *dataset* yang berlabel terdiri dari deskripsi dan spesifikasi label. Tujuan dari *Supervised learning* adalah mengembangkan sistem yang mampu mengestimasi nilai untuk membuat prediksi yang akurat pada situasi yang tidak diberikan dalam rangkaian pelatihan pembelajaran. Dalam suatu kondisi lingkungan tertentu *Supervised learning* tidak memadai untuk belajar dan berinteraksi terlepas dari relevansi dan signifikansinya dalam sebagian besar aplikasi pembelajaran mesin. *Reinforcement learning* berbeda dengan *Supervised learning* karena belajar dari interaksi dan akan melengkapi sistem *Reinforcement learning* [9]. Hal ini menegaskan bahwa *Supervised learning* bukanlah pendekatan terbaik untuk masalah pengoptimalan keuntungan dalam perdagangan valas.

Pasar valas adalah pasar terbesar di dunia yang membuat menarik bagi investor [10]. Dalam artikel ini, kami telah mencoba memecahkan masalah otomatisasi keuntungan dan meminimalkan resiko kerugian di pasar valas dengan menggunakan *Deep Recurrent Q-Learning.*

1.1. Q-learning

Reinforcement Learning [3] adalah sejenis algoritma pembelajaran mesin yang dilakukan oleh agen untuk mempelajari dan meningkatkan kebijakan melalui uji coba dan untuk memperoleh akumulasi hadiah dalam jangka panjang maksimum pada suatu tugas tertentu. Program akan memberi tahu agen tindakan mana yang harus dipilih di setiap keadaan lingkungan untuk memperoleh akumulasi hadiah yang lebih baik. Tugas *reinforcement learning* biasanya digambarkan sebagai Proses Keputusan *Markov* (MDP), yang dapat diwakili oleh sebuah *tuple* (*S,A,P,R* , γ). Dalam *tuple* ini, S adalah himpunan terbatas kondisi, A adalah himpunan terbatas tindakan, P adalah matriks probabilitas transisi kondisi yang dirumuskan sebagai berikut:

(1) $P_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$

R adalah fungsi imbalan hadiah yang diformulasikan sebagai berikut:

(2) $R_s^a = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$

γ adalah faktor diskon dan $\gamma \epsilon [0,1]$

Ketika agen mempelajari kebijakan π di lingkungan tertentu, sebenarnya agen mempelajari distribusi atas tindakan dengan status tertentu:

(3) $\pi(a \mid s) = \mathbb{P}[A_t = a \mid S_t = s]$

Perintah-perintah diberikan π, agen dapat mengetahui tindakan mana yang harus dipilih sesuai kondisi. Untuk memperbaiki aturan π, *reinforcement learning* dapat dibagi menjadi RL berbasis nilai dan RL berbasis perintah [11]. Pembelajaran berbasis nilai memiliki Q-*function* yang mewakili pengembalian nilai yang diharapkan mulai dari keadaan s, mengambil tindakan a, dan kemudian mengikuti perintah π. Jika agen telah mempelajari Q-*function* yang akurat dari lingkungan ini, agen dapat memperoleh akumulasi hadiah terbaik dengan memilih tindakan dengan Q-*value* tertinggi di setiap langkah. Dibandingkan dengan RL berbasis perintah, RL berbasis nilai telah menunjukkan kinerja yang sangat baik dalam tugas-tugas yang memiliki dimensi rendah dan ruang tindakan yang berbeda. Semua karakteristik di atas membuat RL berbasis nilai memenuhi syarat untuk memecahkan masalah perdagangan valas.

Q-*learning* menurut [12] adalah model algoritma berbasis nilai untuk melakukan estimasi. Inti dari Q-*learning* adalah Q-*table* yang mewakili Q-*value* untuk setiap pasangan keadaan dan tindakan pada suatu lingkungan.

(4) $Q_\pi(s, a) = \mathbb{E}_\pi[R_t \mid S_t = s, A_t = a]$

Pada rumus persamaan (4) π adalah perintah keputusan, dan R_t merupakan akumulasi imbalan yang diberikan oleh:

(5) $R_t = \Sigma_{t=0}^{+\infty} \gamma^t r_{t+1}$

Disini γ dapat diartikan sebagai bobot imbalan yang akan datang. Jika $\gamma = 1$, ini menyatakan bahwa agen memperhatikan hadiah yang akan datang sama dengan hadiah langsung. Sebaliknya, jika $\gamma = 0$, ini menunjukkan bahwa agen hanya memperhatikan pada hadiah langsung saja. Agen meningkatkan perintah dengan persamaan optimalitas *Bellman*:

(6) $Q^*(s, a) = \mathbb{E}\left[r + \gamma \max_a Q^*(s', a')\right]$

(7) $Q'(s, a) = Q(s, a) + \alpha(Q^*(s, a) - Q(s, a))$

Dengan memperbaharui Q-*table* secara iteratif, Q-*function* pada akhirnya akan bertemu dengan Q-*function* yang optimal $Q^*(s,a)$ [11].

1.2. Deep Q-learning

Deep Q-learning adalah Q-*learning* dengan Q-*table* diganti dengan jaringan saraf (*deep neural network*). Pendekatan ini bertujuan untuk mengatasi masalah bahwa ketika ruang keadaan dan ruang tindakan berlanjut atau perbedaan tak terhingga, karena iterasi Q-*value* dalam Q-*table* dengan sampel terbatas menjadi tidak mungkin dilakukan. Karena telah dibuktikan bahwa jaringan saraf dengan 3 lapisan mampu mengekspresikan fungsi apa pun [13], gagasan utama deep Q-*learning* adalah menggunakan jaringan saraf untuk mengestimasi Q-*function*. Q-*network* dapat dilatih dengan meminimalkan kesalahan antara Q-*value* yang ditingkatkan pada diberikan oleh rumus (6) dan Q-*value* asli. Fungsi kerugian diformulasikan sebagai berikut:

(8) $L_t(\theta_t) = \mathbb{E}_{s,a,r,s'}\left[\left(y_t - Q_{\theta_t}(s,a)\right)^2\right]$

di mana t waktu saat ini, dan $y_t = r + \gamma \max_a Q_{\theta_t}(s', a')$.

Fungsi kerugian dapat diminimalkan dengan algoritma *stochastic gradient descend* (SGD). Dan *gradien* diberikan oleh:

(9) $\nabla_{\theta_t} L_t(\theta_t) = \mathbb{E}_{s,a,r,s'}\left[\left(y_t - Q_{\theta_t}(s,a)\right)\nabla_{\theta_t} Q_{\theta_t}(s,a)\right]$

1.3. Deep Recurrent Q-Learning

Dalam kerangka kerja DQN, ada anggapan tersembunyi bahwa keadaan yang kita definisikan adalah pengamatan penuh terhadap lingkungan pada setiap langkah waktu. Namun, untuk lingkungan perdagangan valas, variabel yang ada dalam dimensi waktu biasanya tidak dapat ditemukan dengan baik oleh kerangka kerja DQN [14]. Untuk mengatasi masalah ini, kami mereferensikan *Deep Recurrent Q-networks* yang diperkenalkan oleh [15]. Dalam kerangka DRQN, h_t adalah keadaan tersembunyi dari unit LSTM dan merepresentasikan informasi dari langkah waktu sebelumnya. LSTM dapat menemukan informasi yang tersembunyi dalam korelasi sementara dan dapat menyimpan fitur-fitur penting dalam keadaan sebelumnya [10]. Dengan demikian, menerapkan LSTM dalam kerangka kerja DQN dapat membuat keadaan yang kami definisikan lebih dekat dengan pengamatan penuh terhadap lingkungan perdagangan.

METODE

Metode yang digunakan dalam artikel penelitian ini disusun sebagai berikut:

3.1 Pengumpulan Data

Data yang digunakan dalam percobaan ini melibatkan 5 tahun data mulai dari 01/01/2015 sampai 12/31/2020. Database dibagi menjadi *dataset* pelatihan (01/01/2015-12/31/2018) dan *dataset* pengujian (01/01/2019-12/31/2020). Data yang digunakan adalah mata uang EUR/USD untuk harga penutupan harian dalam penelitian ini.

3.2 Persiapan dan Format Data

Semua kumpulan *dataset* dikelompokkan berdasarkan tanggal dan harga. Kumpulan data yang diperoleh pertama-tama diperiksa untuk nilai *null* dan nilai yang hilang. Baris dengan nilai *null* dan nilai yang hilang akan dihapus. Karena data disajikan sebagai nilai deret waktu, bidang tanggal diperlakukan sebagai indeks dan karenanya diubah menjadi objek tanggal untuk kemudahan memanipulasi data. Dataset tersebut kemudian dibagi menjadi dataset pelatihan dan dataset pengujian berdasarkan nilai tanggal tetap yang dipilih. *Dataset* pelatihan terdiri dari nilai yang diindeks lebih rendah

dari nilai tanggal tetap dan *dataset* pengujian terdiri dari nilai yang diindeks lebih tinggi dari nilai tanggal tetap. Latihan ini diterjemahkan menjadi prediksi harga masa depan berdasarkan data pasar historis atau masa lalu.

3.3 Pelatihan Agen

Pertama-tama kami menggunakan sekumpulan kecil *dataset* pelatihan untuk menginisialisasi parameter Q-*network*. Kemudian, pada setiap hari perdagangan, keadaan awal digunakan sebagai masukan dari Q-*network* dan keluarannya adalah nilai-nilai Q untuk tiga tindakan yang berbeda. Agen akan memilih tindakan dengan nilai tertinggi dengan probabilitas $(1 - \varepsilon)$ dan akan mendapatkan imbalan hadiah. Kemudian, sampel perdagangan {s, a, r, s'} akan disimpan dalam buffer. Karena *dataset* harian pelatihan cukup kecil untuk *deep neural network* dan dapat menyebabkan *overfitting*. Untuk memperkaya data yang digunakan untuk memperbarui parameter Q-*network* dan membuat pembaruan lebih stabil, kami menyimpan tiga sampel dengan tiga imbalan hadiah berbeda ke dalam *buffer*. Kemudian selanjutnya, akumulasi hadiah dihitung dan sejumlah sampel ukuran *batch* akan diambil secara acak dari *buffer* untuk memperbarui parameter Q-*network*. Seluruh proses akan berlanjut hingga akhir data pengujian dan akan memperoleh total akumulasi hadiah.

Meskipun *Deep Q-network* dapat mempelajari karakteristik pasar valas secara efektif, masih ada ruang untuk perbaikan. Seperti diketahui bahwa pasar keuangan penuh dengan kebisingan dan ketidakpastian, dan faktor-faktor yang mempengaruhi harga valas dapat berlipat ganda dan akan berubah dari waktu ke waktu. Hal ini membuat proses perdagangan valas lebih seperti proses keputusan *Partially Observable Markov Decision Process* (POMDP) karena keadaan yang kita gunakan tidak sama dengan keadaan lingkungan perdagangan valas yang sebenarnya. Jadi, kami mencoba memanfaatkan karakteristik memori LSTM untuk menemukan fitur yang berubah dari waktu ke waktu, dan kami ingin mengeksplorasi bahwa dengan status yang sama dengan input DQN atau DRQN, LSTM dapat mengintegrasikan informasi yang tersembunyi dalam dimensi waktu. Dengan demikian memungkinkan untuk membuat POMDP lebih dekat dengan MDP [16].

Meskipun *Deep Q-learning* dapat bekerja dengan baik pada banyak tugas, salah satu kelemahan yang diketahui dari *Deep Q-learning* adalah cukup tidak stabil, terutama di lingkungan perdagangan valas yang bising dan waktu yang bervariasi. Jadi kami menggunakan dua pendekatan yang terkenal [17] untuk membuat DQN lebih stabil. Salah satu pendekatannya adalah dengan menggunakan jaringan target terpisah $\tilde{Q}$ untuk menghitung peningkatan Q yang diberikan oleh persamaan *Bellman*, formulasi (6) kemudian diubah menjadi:

$$Q(s,a) = r + \gamma \max_{a} Q(s',a') \quad (10)$$

Kami menerapkan pendekatan yang diperkenalkan [2] untuk memperbarui parameter jaringan target $\tilde{Q}$, yaitu jaringan target diperbarui setiap 100 iterasi dengan parameter jaringan Q asli, sehingga jaringan target $\tilde{Q}$akan lebih lambat memperbarui dari Q-*network* yang asli dan menjadi lebih stabil. Algoritmanya adalah sebagai berikut:

Algorithm Deep Q-learning with Experience Replay

Inisialisasi ulang memori D ke kapasitas N
Inisialisasi jaringan dengan bobot acak θ^Q
Inisialisasi jaringan target dengan bobot

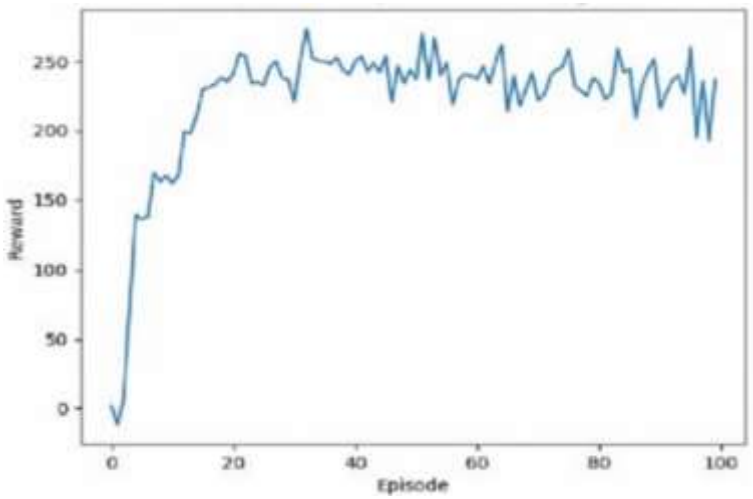
$$\theta^{Q'} \leftarrow \theta^Q$$

for episode = 1, M **do**

for t = 1, T **do**
Dengan probabilitas ε pilih tindakan acak di a_t
jika tidak pilih $a_t = max\ Q(s', a'; \theta)$
Jalankan tindakan a_t dan amati hadiah r_i
Set (s_t, a_t, r_t, S_{t+1}), simpan sampel trading di D
Mencoba transisi mini-batch acak dari D
Tetapkan $y_j = r_j + \gamma maxQ(s_{t+1}, a_{t+1}; \theta)$
Melakukan langkah penurunan gradien pada $\left(y_j - Q(s_j, a_j; \theta)\right)^2$
berdasar persamaan (9)
Perbarui bobot jaringan target:

$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$$

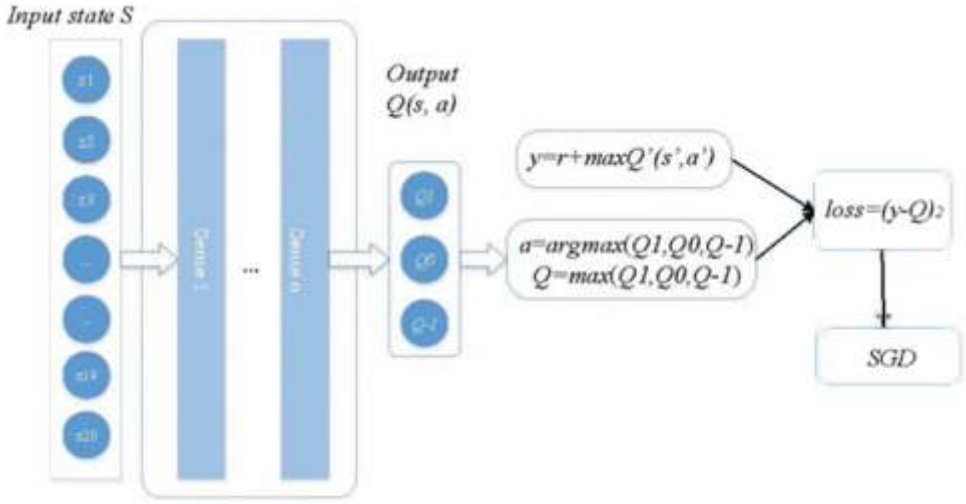
end for end for



Gambar 1. Akumulasi hadiah DQN dari 100 episode pertama pada data pelatihan

3.4 Pengujian Agen

Pertama-tama kami menggunakan data pelatihan untuk menginisialisasi *deep Q-network* dan versi perulangannya sebanyak 30 iterasi, kemudian model kinerja diujikan pada kumpulan *dataset* pengujian. Struktur dasar Q-*network* yang ditampilkan pada Gambar 2.



Gambar 2. Struktur lapisan *Deep Q-network*

Deep Q-network memiliki 4 lapisan dengan jumlah unit untuk setiap lapisan masing-masing yang terdiri dari 20,10,10 dan 3. Kami menggunakan *ReLU* sebagai fungsi aktivasi. Tingkat pembelajaran diatur menurun secara eksponensial setiap 100 langkah pada *dataset* pelatihan dengan nilai tingkat awal adalah 0,001, sedangkan pada *dataset* pengujian, tingkat pembelajaran ditetapkan sebagai nilai tetap 0,001. Untuk bagian Q-*learning*, faktor diskon diatur ke γ = 0.79, nilai ini cukup kecil karena masalah yang kami definisikan adalah proses perdagangan harian jangka pendek sehingga kami lebih memperhatikan imbalan hadiah saat ini. *Max* ε diatur ke 0.99 dan dimulai dengan angka 0. Ukuran *batch* diatur sampai 100 dan parameter untuk target Q-*network* diganti setiap 100 iterasi. Sedangkan untuk model DRQN, time step diatur menjadi 20, dan jumlah unit RNN diatur menjadi 6.

Dalam makalah ini kami menggunakan dua model, yaitu beli, jual, tahan dan DQN acak. Yang pertama berarti membeli satu mata uang pada hari pertama *dataset* pelatihan dan menahan transaksi mata uang tersebut sampai akhir. Dan untuk transaksi DQN acak berarti pada setiap langkah waktu, agen memilih tindakan secara acak terlepas dari nilai Q-*value* untuk setiap tindakan. Parameter yang ditetapkan sama dengan transaksi DQN. Untuk verifikasi bahwa struktur yang digunakan akan meningkatkan bagi transaksi DQN, kami juga mengeksplorasi kinerja DQN dengan jumlah lapisan yang berbeda.

HASIL DAN PEMBAHASAN

Hasil pengujian dari *dataset* yang telah dilatih sebagai berikut :

4.1 *Discount Factor*

Kami mengeksplorasi kinerja DQN dengan lapisan yang berbeda dan hasilnya adalah sebagai berikut. Tabel I, menunjukkan besar hadiah untuk faktor diskon yang berbeda.

Table 1. Besaran hadiah yang sesuai untuk faktor diskon yang berbeda

γ	***0.43***	***0.60***	***0.75***	***0.81***
Besar hadiah	3	5	10	20

Table 2. Akumulasi rata-rata hadiah dengan model DQN

γ	**Jumlah lapisan DQN**			
	Nol	**Satu**	**Dua**	**Tiga**
0.43	-25.6	143.6	162.6	-85.5
0.60	-18.7	157.2	257.6	-79.9
0.75	6.4	190.3	**293.0**	-20.2
0.81	-43.1	209.0	207.9	-99.6

Dari hasil yang ditampilkan pada Tabel 2 kita dapat menyimpulkan bahwa untuk transaksi harian, faktor diskon yang tepat adalah 0.79.

4.2 *Learning Rate*

Tingkat pembelajaran merupakan faktor penting yang mempengaruhi kinerja model. Kami mencoba untuk membandingkan kinerja dari tingkat pembelajaran yang berbeda seperti yang kami tampilkan pada Tabel 3.

Tabel 3. Tingkat pembelajaran yang berbeda berdasar akumulasi hadiah

Learning rate	**0.01**	**0.001**	**Penurunan eksponensial**	**Pengaturan tingkat pembelajaran**
DQN dengan 2 lapisan	-31.3	190.3	212.1	273.6

4.3 Kinerja DQN dan DRQN

Tabel IV menunjukkan kinerja DQN dan DRQN pada *dataset* pengujian. Dibandingkan dengan dua model sebelumnya, kita dapat melihat bahwa pendekatan *deep Q-learning* dapat menemukan fitur secara otomatis dan membuat keputusan yang tepat untuk memperoleh akumulasi imbalan paling banyak dibandingkan strategi lainnya.

Table 4. Akumulasi rata-rata berdasarkan model yang berbeda

	Perbandingan Kinerja			
	Buy & Hold	**DQN Acak**	**DQN**	**DRQN**
Total Rata-rata Profit	159.4	5.0	285.8	338.8

Dapat kita lihat bahwa model DQN dan DRQN mengungguli model Buy & Hold, hal ini menunjukkan bahwa DQN dan DQRN dapat secara efektif mempelajari strategi yang lebih menguntungkan, dan DRQN lebih baik daripada DQN.

KESIMPULAN

Dalam artikel ini kami mengeksplorasi kinerja *deep Q-learning* dan *deep recurrent Q-learning* untuk perdagangan valas. Hasilnya menunjukkan bahwa *deep Q-network* dapat mempelajari pola yang menguntungkan dari data perdagangan valas biasa dan memanfaatkannya untuk mencapai akumulasi hadiah yang tinggi. Karena lingkungan perdagangan valas lebih dekat dengan POMDP, kami juga mengamati bahwa memasukkan pengulangan ke dalam Q-*learning* dapat membawa peningkatan kinerja dalam tugas perdagangan valas. Hasilnya menunjukkan bahwa DRQN adalah metode alternatif untuk membawa perbaikan sistematis.

Di masa mendatang, kami akan mengeksplorasi penerapan pendekatan pembelajaran penguatan lainnya pada tugas perdagangan valas. Perdagangan berbagai aset saham dan komoditas juga merupakan arah opsional untuk dijelajahi.

REFERENSI

[1] U. S. Shanthamallu and A. Spanias, "Introduction to Machine Learning," *Synth. Lect. Signal Process.*, pp. 1–8, 2022, doi: 10.1007/978-3-031-03758-0_1.

[2] P. Treleaven, M. Galas, and V. Lalchand, "Algorithmic Trading Review," *Commun. ACM*, vol. 56, no. 11, pp. 76–85, Nov. 2013, doi: 10.1145/2500117.

[3] A. M. Andrew, "REINFORCEMENT LEARNING: AN INTRODUCTION by Richard S. Sutton and Andrew G. Barto, Adaptive Computation and Machine Learning series, MIT Press (Bradford Book), Cambridge, Mass., 1998, xviii + 322 pp, ISBN 0-262-19398-1, (hardback, £31.95).," *Robotica*, vol. 17, no. 2, pp. 229–235, 1999, doi: DOI: 10.1017/S0263574799211174.

[4] Z. Jiang, D. Xu, and J. Liang, "A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem," *arXiv [q-fin.CP]*, 2017.

[5] V. Mnih *et al.*, "Playing Atari with Deep Reinforcement Learning," *arXiv Prepr.*

arXiv1312.5602, 2013.
[6] J. Carapuço, R. Neves, and N. Horta, "Reinforcement learning applied to Forex trading," *Appl. Soft Comput.*, vol. 73, pp. 783–794, 2018, doi: 10.1016/j.asoc.2018.09.017.
[7] Z. Jiang and J. Liang, "Cryptocurrency portfolio management with deep reinforcement learning," *2017 Intell. Syst. Conf. IntelliSys 2017*, vol. 2018-Janua, pp. 905–913, 2018, doi: 10.1109/IntelliSys.2017.8324237.
[8] M. Hausknecht and P. Stone, "Deep Recurrent Q-Learning for Partially Observable MDPs," 1997.
[9] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, Second edi. Cambridge, Massachusetts: The MIT Press, 2018.
[10] J. Chen, "Forex - FX," *Investopedia*, 2018.
[11] J. Moody, M. Saffell, Y. Liao, and L. Wu, "Reinforcement Learning for Trading Systems and Portfolios: Immediate vs Future Rewards BT - Decision Technologies for Computational Finance: Proceedings of the fifth International Conference Computational Finance," A.-P. N. Refenes, A. N. Burgess, and J. E. Moody, Eds. Boston, MA: Springer US, 1998, pp. 129–140. doi: 10.1007/978-1-4615-5625-1_10.
[12] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992, doi: 10.1007/BF00992698.
[13] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989, doi: https://doi.org/10.1016/0893-6080(89)90020-8.
[14] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep Direct Reinforcement Learning for Financial Signal Representation and Trading," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, no. 3, pp. 653–664, Mar. 2017, doi: 10.1109/TNNLS.2016.2522401.
[15] M. J. Hausknecht and P. Stone, "Deep Reinforcement Learning in Parameterized Action Space," *CoRR*, vol. abs/1511.0, 2015.
[16] G. Lample and D. S. Chaplot, "Playing FPS Games with Deep Reinforcement Learning," in *AAAI Conference on Artificial Intelligence*, 2016.
[17] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015, doi: 10.1038/nature14236.