KAMPUS AKADEMIK PUBLISING
Jurnal Sains Student Research (JSSR)
Vol.4, No.6 Desember 2026
e-ISSN: 3025-9851, p-ISSN : 3025-986X, Hal 293-304
DOI : https://doi.org/10.61722/jssr.v4i6.12856

Perancangan Smart Decision NPC Game Turn-Based Menggunakan MLP Dengan MCTS Sebagai Generator Keputusan

M. Ryan Prasetya*[1], Muhammad Haris[2]
[1,2]Jurusan Teknologi Informasi, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara
*Penulis Korespondensi: [1]prasetyaryan0105@gmail.com, [2]muhammadharis@umsu.ac.id

Abstract. *Turn-based games require Non-Player Characters (NPCs) capable of selecting actions based on changes in game conditions. Simple rule-based decisions can produce predictable behavior, while Monte Carlo Tree Search (MCTS) can generate more directed decisions at the cost of repeated search computation. This study designs a smart decision system using MCTS as a decision generator and Multi-Layer Perceptron (MLP) as a model that approximates its decision distribution. The research includes turn-based game design, limited MCTS implementation, dataset generation through MCTS-versus-MCTS simulations, preprocessing and embedding, and MLP training using KL divergence. The first evaluation used 6,894 samples and the second used 15,288 samples. The results show that MLP can learn some MCTS decision patterns. Average accuracy increased from 57.57% to 62.84% across three learning phases in the first evaluation. However, decision signal readability remained low, with a mean interpretability of -46.31 and an MLP win ratio of 0.45. The weighted evaluation score reached 27.29 and did not meet the criterion of 60. In the second evaluation, the best accuracy was 54.82%, indicating that adding hero perspectives and data did not directly improve generalization. Nevertheless, MLP has a simpler inference mechanism because decisions are generated through a single forward propagation.*
Keywords: *Artificial Intelligence, Monte Carlo Tree Search, Multi-Layer Perceptron, NPC, Turn-Based Game*

Abstrak. Gim turn-based membutuhkan Non-Player Character (NPC) yang mampu memilih aksi berdasarkan perubahan kondisi permainan. Keputusan berbasis aturan sederhana dapat menghasilkan perilaku yang mudah diprediksi, sedangkan Monte Carlo Tree Search (MCTS) mampu menghasilkan keputusan yang lebih terarah dengan konsekuensi kebutuhan komputasi dari proses pencarian berulang. Penelitian ini merancang sistem smart decision menggunakan MCTS sebagai generator keputusan dan Multi-Layer Perceptron (MLP) sebagai model yang mengaproksimasi distribusi keputusan tersebut. Penelitian meliputi perancangan gim turn-based, implementasi MCTS terbatas, pembentukan dataset melalui simulasi MCTS melawan MCTS, preprocessing dan embedding, serta pelatihan MLP menggunakan KL divergence. Evaluasi pertama menggunakan 6.894 sampel dan evaluasi kedua menggunakan 15.288 sampel. Hasil menunjukkan bahwa MLP mampu mempelajari sebagian pola keputusan MCTS. Rata-rata akurasi meningkat dari 57,57% menjadi 62,84% pada tiga fase pembelajaran evaluasi pertama. Namun, keterbacaan sinyal keputusan masih rendah dengan mean interpretability sebesar -46,31 dan win ratio MLP sebesar 0,45. Nilai evaluasi berbobot mencapai 27,29 dan belum memenuhi kriteria 60. Pada evaluasi kedua, akurasi terbaik sebesar 54,82%, sehingga penambahan perspektif hero dan data belum meningkatkan generalisasi secara langsung. Meskipun demikian, MLP memiliki mekanisme inferensi yang lebih sederhana karena keputusan dihasilkan melalui satu proses propagasi maju.
Kata kunci: Kecerdasan Buatan, Monte Carlo Tree Search, Multi-Layer Perceptron, NPC, Gim Turn-Based

1. LATAR BELAKANG

Perkembangan kecerdasan buatan pada gim mendorong penggunaan Non-Player Character (NPC) yang tidak hanya berfungsi sebagai karakter pendukung, tetapi juga sebagai agen yang mampu memberikan tantangan dan merespons kondisi permainan secara dinamis. NPC pada dasarnya merupakan entitas dalam permainan yang tidak dikendalikan langsung oleh pemain dan dapat menjalankan perilaku berdasarkan aturan

Naskah Masuk:11 September 2026; Revisi: 12 September 2026; Diterima: 13 September 2026; Terbit: 29 September 2026

maupun sistem kecerdasan buatan. Kualitas perilaku NPC menjadi salah satu faktor yang dapat memengaruhi pengalaman bermain karena keputusan yang terlalu sederhana dapat membuat pola permainan mudah diprediksi.

Pada gim berbasis strategi dan turn-based, pengambilan keputusan memiliki peran yang lebih penting karena setiap aksi dilakukan secara bergantian dan dapat memengaruhi kondisi pada fase berikutnya. Pemain maupun agen perlu mempertimbangkan kondisi saat ini, kemampuan karakter, sumber daya, serta kemungkinan respons lawan. Dengan demikian, sistem NPC pada lingkungan tersebut membutuhkan mekanisme pengambilan keputusan yang mampu mempertimbangkan konsekuensi dari suatu aksi dan bukan hanya merespons kondisi berdasarkan aturan statis.

Monte Carlo Tree Search (MCTS) merupakan salah satu metode pencarian yang banyak digunakan untuk permasalahan pengambilan keputusan berurutan. MCTS membangun pencarian berbasis pohon melalui proses eksplorasi, ekspansi, simulasi, dan propagasi balik sehingga kualitas suatu aksi dapat diperkirakan melalui statistik hasil simulasi. Mekanisme ini memungkinkan sistem memilih aksi berdasarkan pengalaman eksplorasi terhadap kemungkinan keadaan permainan berikutnya [1], [2]. Kemampuan tersebut menjadikan MCTS sesuai digunakan sebagai acuan pengambilan keputusan pada lingkungan permainan yang mempunyai ruang aksi cukup besar.

Permasalahan utama penggunaan MCTS pada NPC adalah kebutuhan komputasi yang meningkat seiring jumlah simulasi dan kompleksitas ruang pencarian. Pada setiap pengambilan keputusan, MCTS perlu melakukan sejumlah simulasi untuk memperoleh distribusi statistik yang dapat digunakan sebagai dasar keputusan. Penelitian sebelumnya menunjukkan bahwa penggabungan MCTS dengan model neural network dapat menghasilkan kemampuan pencarian yang kuat, tetapi proses gabungan tersebut juga dapat tetap menimbulkan beban komputasi yang besar [3]. Kondisi tersebut membuka peluang untuk memanfaatkan hasil keputusan MCTS sebagai pengetahuan yang dapat dipelajari oleh model neural network sehingga proses pengambilan keputusan pada saat inferensi menjadi lebih sederhana.

Multi-Layer Perceptron (MLP) merupakan arsitektur jaringan saraf tiruan yang memiliki satu atau lebih hidden layer dan mampu memodelkan hubungan nonlinier antara masukan dan keluaran [4], [5]. MLP dapat digunakan untuk melakukan klasifikasi maupun pemetaan pola berbasis data. Penelitian terkait menunjukkan bahwa arsitektur MLP dapat digunakan pada berbagai permasalahan klasifikasi dan pengambilan keputusan, tetapi performanya sangat dipengaruhi oleh representasi data, jumlah data, arsitektur, dan parameter pelatihan [6], [7], [8], [9].

Berdasarkan kondisi tersebut, penelitian ini merancang sistem smart decision pada NPC gim turn-based dengan menempatkan MCTS sebagai teacher yang menghasilkan distribusi keputusan dan MLP sebagai student yang mempelajari serta mengaproksimasi pola keputusan tersebut. Pendekatan ini berbeda dari penggunaan MCTS sebagai komponen pencarian langsung pada saat inferensi. Dalam penelitian ini, pencarian MCTS digunakan terlebih dahulu untuk menghasilkan data keputusan, kemudian informasi tersebut digunakan untuk membentuk model MLP.

Kontribusi penelitian ini terletak pada perancangan pipeline keputusan yang terdiri atas lingkungan gim turn-based, MCTS terbatas sebagai generator keputusan, dataset berbasis distribusi aksi, serta model MLP yang mempelajari distribusi tersebut. Selain mengukur akurasi pembelajaran, penelitian juga mengevaluasi kemampuan model membaca sinyal kondisi permainan dan menguji kemampuan model ketika cakupan perspektif hero serta data pelatihan diperluas. Penelitian mengenai MCTS telah banyak

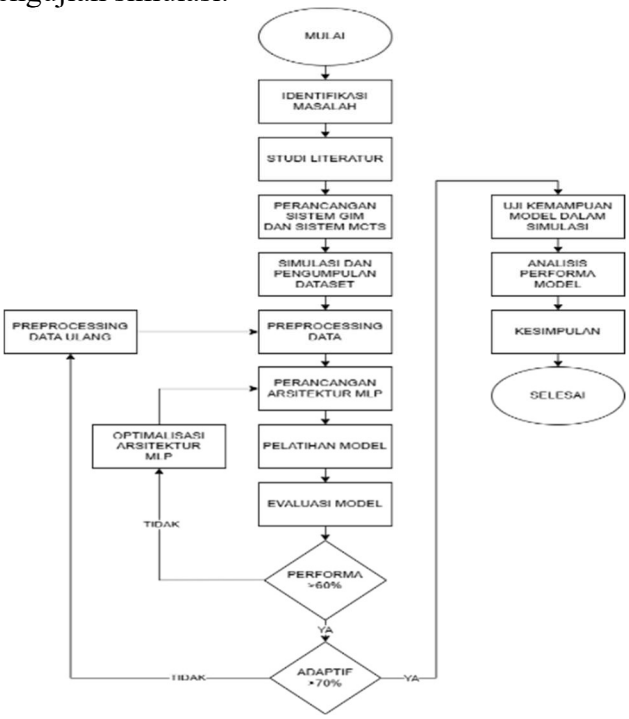
dikembangkan pada berbagai domain, sedangkan pemanfaatan MCTS sebagai generator keputusan untuk kemudian diaproksimasi oleh MLP pada domain gim turn-based masih memiliki ruang penelitian yang cukup besar [3].

Tujuan penelitian adalah merancang dan mengimplementasikan MCTS sebagai generator keputusan pada lingkungan gim turn-based, membangun model MLP yang dapat mempelajari distribusi keputusan MCTS, serta mengevaluasi kemampuan model dalam mempelajari pola keputusan, membaca perubahan kondisi permainan, dan mempertahankan performa ketika menerima data dari perspektif hero yang lebih beragam.

2. METODE PENELITIAN

2.1 Tahapan Penelitian

Penelitian dilakukan melalui beberapa tahapan, yaitu perancangan lingkungan gim turn-based, implementasi MCTS, simulasi untuk pembentukan dataset, preprocessing data, perancangan model MLP, pelatihan model, evaluasi performa, evaluasi adaptabilitas, dan pengujian simulasi.



Gambar 1. Alur Penelitian

Tahapan penelitian dimulai dari identifikasi masalah dan studi literatur. Selanjutnya dibangun lingkungan simulasi gim turn-based yang menjadi domain eksperimen. Sistem tersebut kemudian diintegrasikan dengan MCTS sehingga keputusan agen dapat diperoleh melalui proses pencarian. Setelah sistem berjalan, simulasi MCTS melawan MCTS digunakan untuk menghasilkan data state dan distribusi aksi. Dataset kemudian dinormalisasi dan diubah menjadi representasi numerik yang sesuai untuk pelatihan MLP.

Tahap berikutnya adalah perancangan model yang terdiri atas encoder embedding dan model decision. Model kemudian dilatih menggunakan dataset hasil simulasi. Evaluasi dilakukan melalui pengujian akurasi distribusi aksi, pengujian keterbacaan sinyal pada domain kecil, evaluasi berbasis pertandingan melawan MCTS, dan pengujian adaptabilitas menggunakan dataset dengan cakupan perspektif hero yang lebih luas. Alur

ini mengikuti rancangan penelitian pada skripsi yang menempatkan MCTS sebagai generator dataset dan MLP sebagai model pendekatan keputusan [10].

2.2 Perancangan Gim Turn-Based

Lingkungan eksperimen dibuat secara khusus untuk penelitian. Sistem terdiri atas State, Player, Skill, dan Match. State menyimpan kondisi permainan dari kedua pihak, Player merepresentasikan hero sebagai agen, Skill menyimpan atribut kemampuan, sedangkan Match berfungsi sebagai basis simulasi dan penghubung antara gim dengan sistem MCTS.

Permainan menggunakan lima hero dengan karakteristik berbeda. Setiap hero memiliki lima skill dan satu aksi Guard sehingga terdapat enam kemungkinan aksi pada kondisi permainan. Atribut hero meliputi health, defense, energy, energy regeneration, accuracy, evasion, mastery, dan element. Setiap skill mempunyai atribut offense, defense, element, towards, mastery, dan energy cost.

Aturan utama permainan meliputi penambahan energy pada setiap siklus, pembatasan penggunaan skill berdasarkan energy yang tersedia, aksi Guard sebagai aksi tanpa konsumsi energi, serta kemungkinan critical pada serangan. Game juga memiliki pengaruh atribut element, accuracy, evasion, defense, dan mastery sehingga hasil eksekusi aksi dapat berbeda pada kondisi yang berbeda.

Tabel 1. Atribut Hero

Atribut	**Hero**				
	No.1	**No.2**	**No.3**	**No.4**	**No.5**
Health	770	750	780	710	850
Defense	23	26	28	23	35
Energy	300	310	280	260	340
Element	Fire	Water	Wood	Wind	Earth
Energy Regen	60	57	57	65	45
Accuracy	8	7	6	9	6
Evasion	5	6	4	7	3
Mastery	1.5	1.5	1.5	1.5	1.5

Perbedaan atribut digunakan agar masing-masing hero mempunyai karakteristik keputusan yang berbeda. Selain itu, atribut skill disusun untuk memberikan variasi pilihan berupa aksi offense, defense, support, dan Guard. Variasi tersebut dibutuhkan agar model tidak hanya mempelajari satu pola tindakan yang dominan.

2.3 Implementasi MCTS

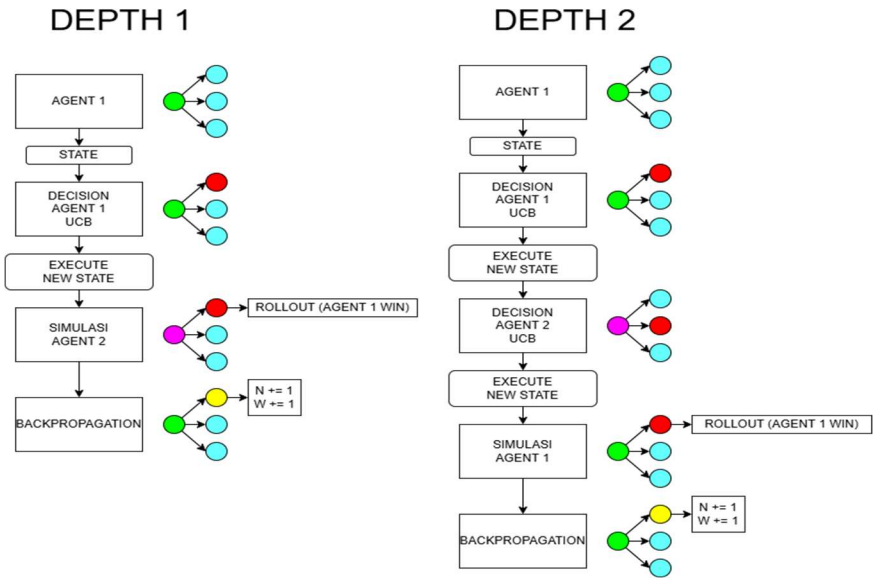
MCTS pada penelitian ini menggunakan pendekatan limited-depth MCTS, bukan pencarian MCTS standar dengan kedalaman yang tidak dibatasi. Pencarian dibatasi hingga dua kedalaman dan selanjutnya dilanjutkan dengan rollout sampai diperoleh hasil simulasi. Pendekatan ini dipilih karena lingkungan gim bersifat tidak sepenuhnya deterministik akibat adanya faktor acak pada eksekusi aksi.

Secara umum, MCTS terdiri atas tahap eksplorasi, ekspansi, simulasi, dan propagasi balik [1], [2]. Pada penelitian ini, pemilihan aksi menggunakan Upper Confidence Bound (UCB). Formula UCB yang digunakan adalah:

$$UCB_i = \frac{W_i}{N_i} + c\sqrt{\frac{\ln N}{N_i}} \quad (1)$$

dengan Wi merupakan total hasil simulasi pada aksi ke-i, Ni merupakan jumlah kunjungan aksi ke-i, N adalah jumlah total kunjungan pada node, dan c merupakan konstanta eksplorasi.

Pada setiap pengambilan keputusan, MCTS menerima daftar aksi yang tersedia berdasarkan jumlah energy. Setiap aksi kemudian dievaluasi melalui simulasi berulang. Statistik kunjungan dan hasil simulasi digunakan untuk membentuk distribusi keputusan. Penelitian menggunakan 1.000 iterasi per pengambilan keputusan, sedangkan pertandingan dibatasi sekitar 20–24 turn untuk menghindari simulasi yang mengalami atrisi terlalu panjang.



Gambar 2. Struktur MCTS Terbatas

Output utama MCTS yang digunakan dalam pembentukan dataset bukan hanya aksi terpilih, tetapi distribusi kunjungan terhadap seluruh aksi yang tersedia. Dengan demikian, model MLP mempelajari kecenderungan pilihan MCTS dan bukan hanya satu label aksi.

2.4 Pembentukan dan Preprocessing Dataset

Dataset dibentuk melalui simulasi MCTS melawan MCTS pada berbagai kombinasi hero. Setiap state yang diperoleh selama simulasi disimpan bersama distribusi keputusan MCTS. Data diambil dari sudut pandang kedua agen sehingga state dari sisi P2 dapat ditukar menjadi perspektif P1. Dengan demikian, model mempelajari hubungan state–keputusan secara konsisten dari sudut pandang agen yang sedang mengambil keputusan. Data state terdiri atas fitur dinamis dan fitur statis dari kedua player. Fitur dinamis mencakup health, defense, energy, dan guard, sedangkan fitur statis mencakup atribut dan identitas hero serta atribut skill.

Nilai numerik dinormalisasi untuk menjaga skala fitur berada pada rentang yang lebih sesuai untuk model. Sebagai contoh, health dinormalisasi berdasarkan nilai maksimum yang digunakan sebagai batas. Fitur kategori seperti element dan towards direpresentasikan dengan one-hot encoding. Data keluaran berupa distribusi kunjungan MCTS kemudian dinormalisasi sehingga jumlah seluruh probabilitas aksi bernilai satu. Dengan representasi tersebut, label tidak lagi hanya menunjukkan satu aksi benar, tetapi memberikan informasi mengenai seberapa kuat masing-masing aksi dipertimbangkan oleh MCTS.

Dalam evaluasi pertama digunakan 6.894 sampel yang berasal dari perspektif hero pada kombinasi pertarungan yang digunakan untuk eksplorasi arsitektur. Pada evaluasi kedua, cakupan diperluas sehingga digunakan 15.288 sampel dari perspektif hero yang lebih beragam.

2.5 Perancangan Model MLP

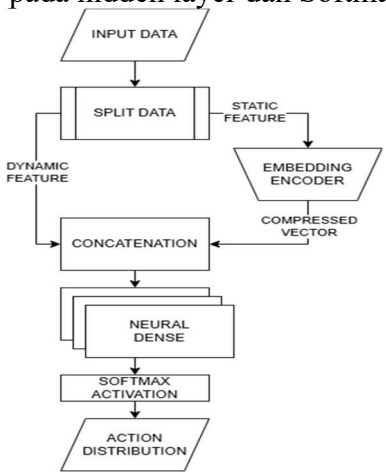
Model terdiri atas dua komponen utama, yaitu Embedding Encoder dan Decision Model. Embedding Encoder digunakan untuk mengubah representasi fitur statis hero menjadi vektor yang lebih ringkas. Hasil embedding kemudian digabungkan dengan fitur dinamis sebelum diteruskan ke Decision Model. Encoder menggunakan struktur:

Input(90)
Linear(64)
ReLU
Linear(32)
ReLU
Linear(20)

Representasi tersebut kemudian digabung dengan fitur dinamis sebagai masukan Decision Model. Arsitektur utama Decision Model yang digunakan dalam eksperimen adalah:

Input(44)
Linear(64) Relu
Linear(32) Relu
Linear(32) Relu
Linear(16) Relu
Linear(6) Softmax

dengan fungsi aktivasi ReLU pada hidden layer dan Softmax pada output layer.



Gambar 3. Struktur Arsitektur MLP

Output sebanyak enam neuron merepresentasikan Guard dan lima skill. Fungsi Softmax digunakan agar keluaran model berupa distribusi probabilitas yang dapat dibandingkan dengan distribusi aksi yang dihasilkan MCTS.

Pelatihan dilakukan menggunakan KL Divergence sebagai fungsi loss karena target yang digunakan berbentuk distribusi. Pada eksplorasi arsitektur digunakan tiga fase learning rate yang dilatih bertahap, yaitu 1e-3, 1e-4, dan 1e-5 sedangkan batch size dibandingkan pada nilai 16, 32, dan 64. Setiap fase menggunakan 50 epoch.

2.6 Metode Evaluasi

Evaluasi dilakukan pada beberapa tingkat. Evaluasi pertama bertujuan mengukur kemampuan model dalam mempelajari distribusi keputusan MCTS. Pengujian menggunakan sampel acak dari dataset dan memberikan skor 1 ketika keputusan model sesuai dengan target, skor 0,5 ketika keputusan berada satu posisi dari target, dan skor 0 pada kondisi lainnya.

Selain akurasi, dilakukan pengujian keterbacaan sinyal. Pengujian ini digunakan untuk mengetahui apakah model merespons perubahan kondisi permainan pada variabel seperti health, energy, defense, dan guard. Pengujian dilakukan pada domain kecil sehingga pengaruh perubahan variabel dapat diamati secara lebih terkontrol.

Model juga diuji dalam simulasi melawan MCTS. Dalam evaluasi tersebut, MCTS diposisikan sebagai teacher sedangkan MLP diposisikan sebagai student. Oleh karena karakteristik algoritma berbeda, hasil pertandingan tidak digunakan sebagai perbandingan langsung 1:1 antara dua algoritma yang setara. MCTS melakukan pencarian secara realtime, sedangkan MLP menghasilkan keputusan berdasarkan pola yang telah dipelajari.

Untuk menggabungkan hasil akurasi, interpretability, dan win rate, penelitian menggunakan rata-rata berbobot, di mana interpretability diberi bobot berdasarkan win rate MLP sehingga digunakan persamaan.

$$Score = \frac{Accuracy + (Interpretability \times WinRate)}{1 + WinRate} \quad (2)$$

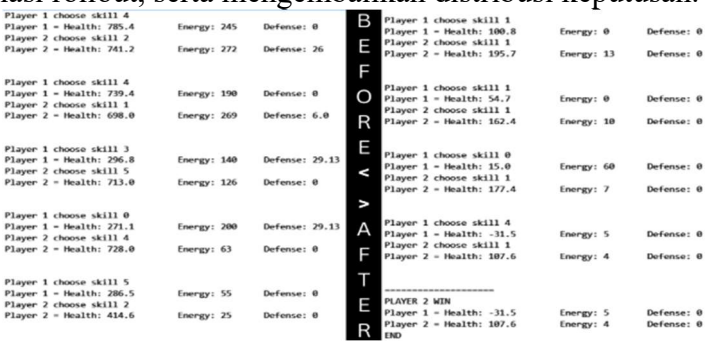
Nilai tersebut merupakan rata-rata berbobot yang dirancang khusus dalam penelitian ini, bukan metrik standar universal. Score merupakan skor dari hasil evaluasi performa, accuracy merupakan rata rata akurasi kemampuan model mempelajari distribusi keputusan, interpretability merupakan performa keterbacaan sinyal, sedangkan win rate merupakan rasio kemenangan model MLP terhadap MCTS sebagai bobot yang merefleksikan hasil kompetitif model ketika berhadapan dengan MCTS.

Evaluasi kedua dilakukan dengan memperluas data menjadi 15.288 sampel untuk mengetahui apakah peningkatan jumlah perspektif hero dan data dapat meningkatkan kemampuan generalisasi model.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Implementasi Sistem MCTS dan Gim

Implementasi sistem berhasil menjalankan simulasi gim turn-based dan mengintegrasikan proses pengambilan keputusan menggunakan MCTS terbatas. Sistem mampu menjalankan state, menentukan aksi yang tersedia berdasarkan energy, melakukan simulasi rollout, serta mengembalikan distribusi keputusan.



Gambar 4. Contoh simulasi gim turn-based yang dikendalikan MCTS

Simulasi menunjukkan bahwa MCTS dapat menghasilkan keputusan berdasarkan kondisi permainan yang berubah. Pada kondisi tertentu, distribusi keputusan beberapa aksi dapat terlihat relatif dekat. Hal tersebut menunjukkan bahwa lebih dari satu aksi

dapat mempunyai peluang hasil yang tidak terlalu berbeda. Kondisi ini menjadi salah satu alasan penggunaan distribusi keputusan sebagai label dibandingkan hard label tunggal.

Keberadaan critical chance dan beberapa faktor acak pada eksekusi aksi menyebabkan hasil simulasi tidak selalu deterministik. Dengan demikian, state yang sama dapat menghasilkan hasil akhir yang berbeda ketika simulasi dijalankan kembali. Kondisi tersebut menjadi salah satu alasan pembatasan kedalaman pencarian MCTS serta penggunaan jumlah iterasi tertentu dalam proses pembentukan dataset.

3.2 Hasil Pelatihan dan Eksplorasi Arsitektur MLP

Eksperimen pertama menggunakan 6.894 sampel dari perspektif hero yang menjadi fokus evaluasi awal. Beberapa konfigurasi arsitektur dibandingkan dengan variasi jumlah neuron, batch size, dan learning rate. Hasil menunjukkan bahwa arsitektur yang relatif sederhana menghasilkan loss yang lebih rendah dibandingkan beberapa arsitektur yang lebih kompleks.

Tabel 2. Sebagian ringkasan hasil loss arsitektur MLP

Arsitektur	Learning Rate	Batch Size 16	Batch Size 32	Batch Size 64
64-32-32-16-6	1e-3	0,0587	0,0615	0,0755
64-32-32-16-6	1e-4	0,0509	0,0539	0,0709
64-32-32-16-6	1e-5	0,0500	0,0531	0,0700
64-32-16-16-6	1e-5	0,0532	0,0533	0,0686
64-32-32-16-6*	1e-5	0,0494	0,0594	0,0687

Loss terendah pada eksplorasi arsitektur berada pada konfigurasi sederhana dengan batch size 16. Hasil tersebut menunjukkan bahwa peningkatan kompleksitas arsitektur tidak selalu memberikan keuntungan pada domain penelitian. Dalam dataset yang relatif terbatas dan mempunyai distribusi keputusan yang ambigu, model dengan terlalu banyak parameter dapat lebih mudah mempelajari distribusi yang dominan dibandingkan pola kondisi secara umum.

Hasil ini sejalan dengan karakteristik MLP yang sangat bergantung pada representasi input dan kualitas data. Penambahan jumlah parameter tanpa dukungan data yang cukup dapat meningkatkan kemampuan model menghafal pola tertentu, tetapi belum tentu meningkatkan kemampuan generalisasi [4], [6].

3.3 Hasil Evaluasi Distribusi Aksi

Evaluasi dilakukan melalui tiga fase pelatihan. Fase pertama menggunakan pelatihan dengan learning rate 1e-3, fase kedua 1e-4, dan fase ketiga 1e-5. Masing-masing fase menggunakan 50 epoch.

Tabel 3. Rata-rata akurasi pada tiga fase pembelajaran

Sampel	Tahap 1	Tahap 2	Tahap 3
100 Sampel	58,10%	61,35%	63,60%
200 Sampel	57,12%	59,46%	62,10%
300 Sampel	57,46%	60,67%	63,58%
400 Sampel	57,01%	60,36%	62,56%
500 Sampel	58,16%	60,25%	62,56%
Mean Total	57,57%	60,42%	62,84%

Mean total meningkat dari 57,57% pada tahap pertama menjadi 60,42% pada tahap kedua dan 62,84% pada tahap ketiga. Hasil tersebut menunjukkan bahwa model berhasil mempelajari pola distribusi yang terdapat pada data. Namun, peningkatan akurasi belum

dapat langsung diartikan sebagai peningkatan kualitas keputusan pada permainan karena model masih perlu diuji pada perubahan state dan interaksi langsung dengan agen MCTS.

Temuan tersebut menjadi penting karena target model bukan hanya menghasilkan prediksi yang dekat dengan label, tetapi juga mampu menghasilkan keputusan yang masuk akal ketika kondisi permainan berubah. Oleh sebab itu, akurasi digunakan sebagai salah satu indikator pembelajaran dan bukan satu-satunya dasar penilaian.

3.4 Evaluasi Keterbacaan Sinyal dan MLP Melawan MCTS

Evaluasi berikutnya dilakukan untuk melihat apakah MLP mampu membaca perubahan kondisi permainan. Pengujian mempertimbangkan variabel health, energy, defense, dan guard. Model diuji pada kondisi yang sedikit dimodifikasi untuk mengetahui apakah perubahan tersebut memengaruhi keputusan yang dikeluarkan.

```
Data Asli
TRUE action: tensor(4)
TRUE Probabilities: tensor([0.0701, 0.0687, 0.0887, 0.0805, 0.6920, 0.0000])

PREDICTED action: 4
PREDICTED: tensor([0.2383, 0.1998, 0.1390, 0.1312, 0.2629, 0.0288])

Variabel G_Health diubah menjadi 10% [Skill 2 Hero adalah Healing]
tensor([0.1000, 0.0000, 0.1150, 0.0000])
tensor([0.8618, 0.0000, 0.1050, 0.0000])
PREDICTED action: 4
PREDICTED: tensor([0.2208, 0.1755, 0.1497, 0.1470, 0.2590, 0.0480])
```

Gambar 5. Contoh evaluasi domain kecil terhadap keputusan MLP

Hasil pengujian menunjukkan bahwa meskipun accuracy distribusi meningkat, model belum konsisten dalam membaca sinyal kondisi permainan. Pada evaluasi pertandingan, MLP diuji melawan MCTS sebagai teacher. Rata-rata hasil pengujian menghasilkan win ratio sekitar 0,45 dan mean interpretability sebesar -46,31.

Mean Accuracy:	60,42
Mean Interpretability:	-46,31
MLP Win Ratio:	0,45
Nilai Evaluasi Berbobot:	27,29
Kriteria Penelitian:	60

Nilai evaluasi berbobot dihitung menggunakan Persamaan evaluasi nilai:

$$Score = \frac{60.42 \ + \ -46.31 \ \times 0.45}{1 + 0.45}$$

$$Score \approx 27.29$$

Skor tersebut belum memenuhi kriteria 60 yang ditetapkan pada penelitian. Hasil ini menunjukkan adanya perbedaan antara kemampuan model mencocokkan pola distribusi pada dataset dengan kemampuan model menghasilkan perilaku keputusan yang baik ketika berhadapan langsung dengan lingkungan simulasi.

Pada beberapa percobaan, MLP tetap mampu memenangkan pertandingan melawan MCTS dengan jumlah iterasi tertentu. Hasil tersebut menunjukkan bahwa model tidak sepenuhnya gagal dalam mempelajari pola keputusan MCTS. Namun, kemenangan individual tidak cukup untuk menunjukkan bahwa MLP memiliki kemampuan keputusan yang setara dengan MCTS karena MCTS melakukan pencarian secara realtime dan kualitas keputusannya dipengaruhi oleh jumlah simulasi.

Dengan demikian, posisi MCTS sebagai teacher dan MLP sebagai student perlu dipertahankan dalam interpretasi hasil. Kekalahan MLP terhadap MCTS tidak dapat

langsung diperlakukan sebagai kegagalan metode secara absolut karena kedua algoritma memiliki mekanisme keputusan yang berbeda.

3.5 Analisis Pengaruh Data dan Perspektif Hero

Setelah evaluasi awal, cakupan dataset diperluas menjadi 15.288 sampel dengan perspektif hero yang lebih beragam. Tujuannya adalah mengetahui apakah peningkatan jumlah data dapat meningkatkan generalisasi model.

Tabel 4. Rata-rata akurasi model pada dataset 15.288 sampel

Sampel	Tahap 1	Tahap 2	Tahap 3
100 Sampel	49,95%	53,30%	54,40%
200 Sampel	51,01%	55,09%	56,23%
300 Sampel	50,06%	55,01%	55,24%
400 Sampel	52,42%	53,76%	54,03%
500 Sampel	50,96%	54,24%	54,18%
Mean Total	50,88%	54,28%	54,82%

Berbeda dari evaluasi pertama, perluasan data justru menghasilkan mean accuracy terbaik sebesar 54,82%. Nilai tersebut lebih rendah dibandingkan mean akurasi 62,84% yang diperoleh pada tahap ketiga evaluasi awal.

Hasil tersebut menunjukkan bahwa penambahan jumlah data dan perspektif hero belum memberikan peningkatan generalisasi secara langsung. Salah satu penyebab yang terlihat dari karakteristik dataset adalah meningkatnya variasi kondisi yang harus dipelajari oleh model. Data tambahan tidak selalu menyediakan pola yang lebih mudah dipelajari, tetapi juga menambah keberagaman distribusi keputusan.

Hal ini menunjukkan bahwa kualitas dataset menjadi faktor penting. Penambahan volume data perlu disertai representasi kondisi yang seimbang dan mampu menggambarkan variasi keputusan secara konsisten. Temuan tersebut sejalan dengan karakteristik model neural network yang sangat dipengaruhi oleh kualitas representasi dan distribusi data [4], [6].

3.6 Pembahasan Pendekatan Teacher–Student

Hasil penelitian memberikan gambaran bahwa penggunaan MCTS sebagai teacher berhasil menyediakan pengetahuan keputusan yang dapat dipelajari oleh MLP. Hal tersebut terlihat dari meningkatnya akurasi pada proses pembelajaran dan kemampuan MLP untuk memenangkan sebagian simulasi melawan MCTS.

Namun demikian, proses aproksimasi menyebabkan sebagian informasi keputusan hilang. MCTS menentukan keputusan melalui proses pencarian berulang terhadap kemungkinan masa depan, sedangkan MLP hanya memetakan input state ke distribusi keluaran berdasarkan pengetahuan yang diperoleh selama pelatihan. Dengan demikian, MLP tidak melakukan eksplorasi baru ketika menerima state yang belum representatif dalam dataset.

Fenomena ini menjelaskan mengapa model dapat memperoleh loss yang relatif rendah tetapi menghasilkan perilaku yang belum konsisten dalam simulasi. Distribusi target MCTS juga tidak selalu memiliki satu aksi dengan keunggulan mutlak karena beberapa aksi dapat mempunyai hasil simulasi yang relatif dekat. Oleh karena itu, pendekatan berbasis distribusi tetap relevan, tetapi membutuhkan representasi data dan proses pelatihan yang mampu mempertahankan informasi keputusan tersebut.

Penelitian mengenai MCTS juga menunjukkan bahwa performa pencarian berkaitan erat dengan kemampuan mengeksplorasi ruang keputusan, sedangkan penelitian neural MCTS menunjukkan adanya peluang untuk memanfaatkan model pembelajaran sebagai pengarah atau pendekatan terhadap proses pencarian [2], [3]. Penelitian ini mengambil arah yang berbeda dengan menggunakan MCTS terlebih dahulu sebagai sumber keputusan, kemudian mempelajari hasilnya menggunakan MLP.

3.7 Implikasi Hasil terhadap Efisiensi Komputasi

Kelebihan utama pendekatan MLP bukan terletak pada kemampuannya mengungguli MCTS secara kualitas keputusan, tetapi pada karakteristik proses inferensinya. MCTS membutuhkan pencarian dan simulasi berulang pada setiap pengambilan keputusan. Sebaliknya, setelah model MLP selesai dilatih, keputusan dapat diperoleh melalui satu proses propagasi maju.

Dengan demikian, penelitian ini menunjukkan adanya potensi trade-off antara kualitas keputusan dan efisiensi komputasi. MLP pada hasil penelitian belum dapat menggantikan MCTS sebagai sumber keputusan berkualitas tinggi, tetapi dapat dipertimbangkan sebagai pendekatan approximator yang lebih ringan ketika kebutuhan inferensi menjadi faktor utama.

Hasil ini juga menunjukkan bahwa peningkatan efisiensi tidak cukup dicapai dengan menambahkan data. Perbaikan pada penelitian berikutnya perlu diarahkan pada kualitas target MCTS, representasi state, keseimbangan distribusi aksi, arsitektur model, dan strategi pelatihan.

4. KESIMPULAN DAN SARAN

Penelitian ini berhasil merancang dan mengimplementasikan sistem smart decision pada NPC gim turn-based dengan menggunakan MCTS terbatas sebagai generator keputusan dan MLP sebagai model yang mengaproksimasi distribusi keputusan tersebut. MCTS berhasil menghasilkan distribusi aksi melalui simulasi berulang yang kemudian digunakan sebagai dataset pelatihan MLP.

Pada evaluasi pertama menggunakan 6.894 sampel, MLP menunjukkan peningkatan kemampuan mempelajari pola keputusan. Mean accuracy meningkat dari 57,57% pada tahap pertama menjadi 62,84% pada tahap ketiga. Namun, peningkatan akurasi tersebut belum diikuti oleh kemampuan membaca sinyal keputusan secara konsisten. Mean interpretability sebesar -46,31 dan win ratio sebesar 0,45 menghasilkan nilai evaluasi berbobot sebesar 27,29, sehingga belum memenuhi kriteria 60 yang digunakan dalam penelitian.

Pada evaluasi kedua menggunakan 15.288 sampel, mean accuracy terbaik mencapai 54,82%. Hasil tersebut menunjukkan bahwa penambahan jumlah data dan perspektif hero belum secara langsung meningkatkan generalisasi model. Kondisi ini menunjukkan bahwa peningkatan jumlah data perlu disertai peningkatan kualitas, keseimbangan, keberagaman, serta representasi data.

Meskipun belum mencapai kualitas keputusan yang setara dengan MCTS, MLP tetap menunjukkan kemampuan untuk mempelajari sebagian pola keputusan yang dihasilkan MCTS dan memenangkan sebagian simulasi melawan MCTS. Selain itu, MLP mempunyai karakteristik inferensi yang lebih sederhana karena keputusan diperoleh melalui satu proses propagasi maju, sedangkan MCTS membutuhkan proses pencarian dan simulasi berulang.

Penelitian selanjutnya disarankan untuk meningkatkan kualitas dataset dengan menyimpan informasi tambahan dari proses pencarian MCTS, seperti nilai evaluasi, reward, jumlah kunjungan, dan karakteristik jalur keputusan. Representasi state juga

dapat dikembangkan agar informasi temporal dan hubungan antarvariabel dapat dipertahankan dengan lebih baik. Selain itu, pendekatan lain seperti model neural yang lebih sesuai untuk representasi state berurutan dapat diuji untuk meningkatkan kemampuan generalisasi dan mendekati kualitas keputusan teacher.

DAFTAR REFERENSI

[1] L. Kocsis and C. Szepesvári, "Bandit based Monte-Carlo Planning," in *Machine Learning: ECML 2006 - 17th European Conference on Machine Learning, Proceedings*, Springer, 2006, pp. 282–293. doi: 10.1007/11871842_29.

[2] M. Świechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk, "Monte Carlo Tree Search: a review of recent modifications and applications," *Artif. Intell. Rev.*, vol. 56, no. 3, pp. 2497–2562, 2023, doi: 10.1007/s10462-022-10228-y.

[3] M. Kemmerling, D. Lütticke, and R. H. Schmitt, "Beyond games: a systematic review of neural Monte Carlo tree search applications," *Appl. Intell.*, vol. 54, no. 1, pp. 1020–1046, 2024, doi: 10.1007/s10489-023-05240-w.

[4] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Comput. Sci.*, vol. 2, p. 420, 2021, doi: 10.1007/s42979-021-00815-1.

[5] K.-L. Du, C.-S. Leung, W. H. Mow, and M. N. S. Swamy, "Perceptron: Learning, Generalization, Model Selection, Fault Tolerance, and Role in the Deep Learning Era," *Mathematics*, vol. 10, no. 24, p. 4730, 2022, doi: 10.3390/math10244730.

[6] M. F. Alam, A. Nuryaman, P. H. Khotimah, A. Parlina, and A. Sihombing, "Optimizing Multi-Layer Perceptron performance in sentiment classification through neural network feature extraction," *BACA J. Dokumentasi dan Inf.*, vol. 46, no. 1, pp. 1–14, 2025, doi: 10.55981/j.baca.2025.8240.

[7] S. H. Gulo and A. H. Lubis, "Penerapan Multi-Layer Perceptron untuk Mengklasifikasi Penduduk Kurang Mampu," *Explor. J. Comput. Sci. Inf. Technol.*, vol. 4, no. 2, pp. 51–59, 2024, doi: 10.47065/explorer.v4i2.1146.

[8] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958, doi: 10.1037/h0042519.

[9] S. Agahian and T. Akan, "Battle royale optimizer for training multi-layer perceptron," *Evol. Syst.*, vol. 13, no. 4, pp. 563–575, 2021, doi: 10.1007/s12530-021-09401-5.

[10] M. Świechowski and D. Ślęzak, "Monte Carlo Tree Search as an Offline Training Data Generator for Decision-Tree-Based Game Agents," 2022. doi: 10.2139/ssrn.4152772.