BAB II
LANDASAN TEORI

2.1 Pengantar Kecerdasan Buatan

2.1.1 Definisi kecerdasan buatan *(AI)*

Gambar 2. 1 Ilustrasi *Artificial Intelligence*

(**Sumber:** *https://www.aratek.co/news/how-artificial-intelligence-ai-is-used-in-biometrics*)

Kecerdasan Buatan (*Artificial Intelligence* atau *AI*) adalah sebuah bidang yang berkembang dalam ilmu komputer yang bertujuan untuk membuat sistem komputer atau mesin yang mampu melakukan berbagai tugas yang, jika dilakukan oleh manusia, memerlukan kecerdasan. Bidang ini mencakup berbagai disiplin ilmu dan teknik yang dirancang untuk memungkinkan mesin memahami, belajar, dan mengambil keputusan secara mandiri. Misalnya, pembelajaran mesin (*machine learning*) adalah salah satu cabang utama *AI* yang memungkinkan

sistem mempelajari pola dari data dan membuat prediksi atau keputusan berdasarkan data tersebut.

Selain itu, *AI* mencakup pemrosesan bahasa alami (*natural language processing*), yang memungkinkan mesin memahami dan memproses bahasa manusia, serta visi komputer (*computer vision*), yang bertujuan membuat mesin mengenali dan menafsirkan gambar atau video seperti yang dilakukan manusia. Kemampuan *AI* untuk mengatasi berbagai masalah kompleks melalui algoritma yang dirancang untuk memecahkan masalah spesifik atau untuk menemukan solusi dari persoalan yang rumit menjadikannya alat yang sangat berguna dalam berbagai industri, mulai dari kesehatan, pendidikan, hingga sektor ekonomi dan teknologi (Kalsum, 2022).

2.1.2 Ruang Lingkup Kecerdasan Buatan *(AI)*

a. Machine Learning

Cabang *AI* yang memungkinkan komputer belajar dari data dan meningkatkan kinerjanya seiring waktu tanpa pemrograman eksplisit adalah pembelajaran mesin (*machine learning*). Teknik ini memungkinkan sistem mengenali pola, membuat prediksi, dan mengambil keputusan berdasarkan data. Dengan semakin banyak data yang diolah, kinerja sistem dalam menyelesaikan tugas akan meningkat secara otomatis, menjadikannya solusi untuk beragam aplikasi, seperti pengenalan gambar, analisis prediktif, dan asisten virtual.

b. *Natural Language Processing (NLP)*

Studi tentang interaksi antara komputer dan bahasa manusia disebut pemrosesan bahasa alami (*natural language processing* atau *NLP*). Bidang ini memungkinkan komputer untuk memahami, menerjemahkan, dan merespons bahasa manusia, baik dalam bentuk teks maupun suara. *NLP* mencakup berbagai teknik seperti analisis sintaksis, analisis semantik, dan pemahaman konteks untuk memungkinkan komunikasi yang lebih alami antara manusia dan mesin, seperti dalam aplikasi asisten virtual, penerjemahan otomatis, dan *chatbot*.

c. *Computer Vision*

Menggunakan algoritma dan teknologi untuk memungkinkan mesin untuk melihat, memahami, dan menginterpretasikan visual seperti gambar dan video.

d. Robotika dan Otomasi

Penerapan *AI* dalam pengembangan robot dan sistem otomasi untuk menyelesaikan tugas-tugas fisik atau intelektual.

e. Pengambilan Keputusan Otomatis

Mengembangkan sistem yang dapat memahami lingkungan mereka dan membuat keputusan yang adaptif dan cerdas berdasarkan situasi yang berubah.

f. Sistem Cerdas dan Pengenalan Pola

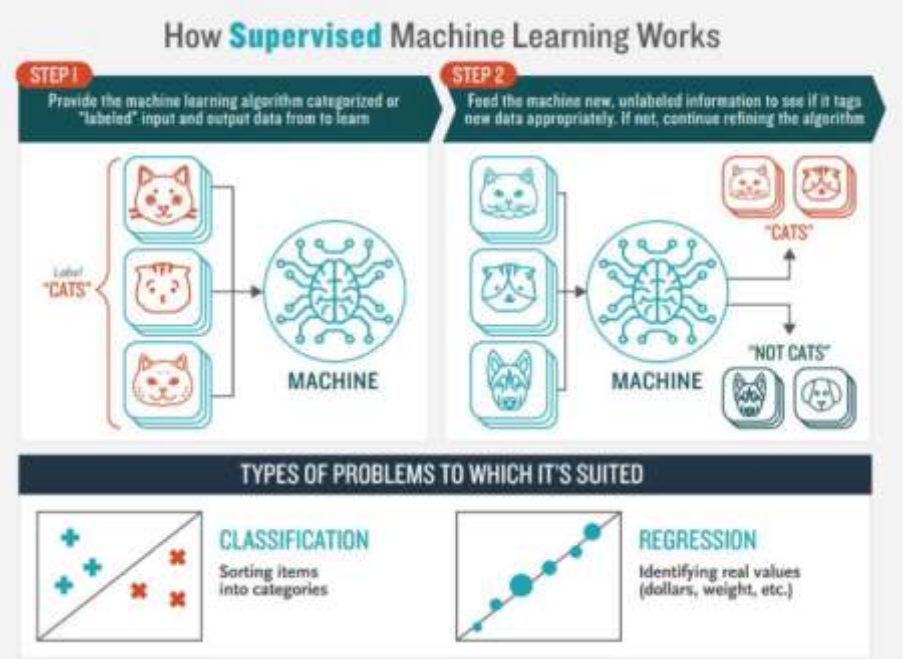
Penggunaan *AI* dalam mengidentifikasi pola kompleks dalam data, seperti dalam pengenalan wajah, prediksi tren, atau analisis data.

2.2 Machine *Learning*

Machine Learning merupakan implementasi dari *Artificial Intelligence.* Ini adalah cara bagi komputer untuk mengasimilasi data atau keadaan dan memberikan solusi atau respon yang dipelajari (Lanham, Machine Learning, 2018). Secara umum *machine learning* terbagi menjadi tiga kategori :

2.2.1 Supervised Learning

Supervised Learning digunakan dalam memprediksi pola yang sudah memiliki contoh data yang lengkap. Label di tiap datanya merupakan *tag* dari data yang ditambahkan dalam model *machine learning* (Roihan, Sunarya, & Rafika, 2020).

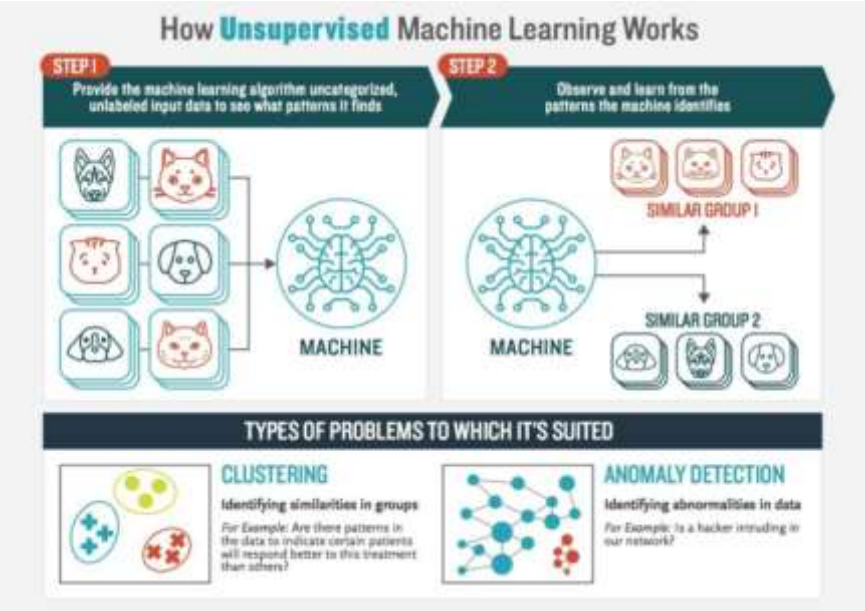


Gambar 2. 2 Cara Kerja *Supervised Learning*

(**Sumber:** *https://www.uc.ac.id/ict/perbedaan-supervised-learning-and-unsupervised-learning/*)

2.2.2 Unsupervised Learning

Unsupervised learning tidak menggunakan label dalam memprediksi target feautures / variable, melainkan menggunakan kesamaan dari atribut-atribut yang dimiliki. Jika atribut dan sifat dari data *feature* yang diekstrak memiliki kemiripan, maka akan dikelompokan (*clustering*), sehingga hal ini akan menimbulkan kelompok (*cluster*) (Ciputra, 2019).

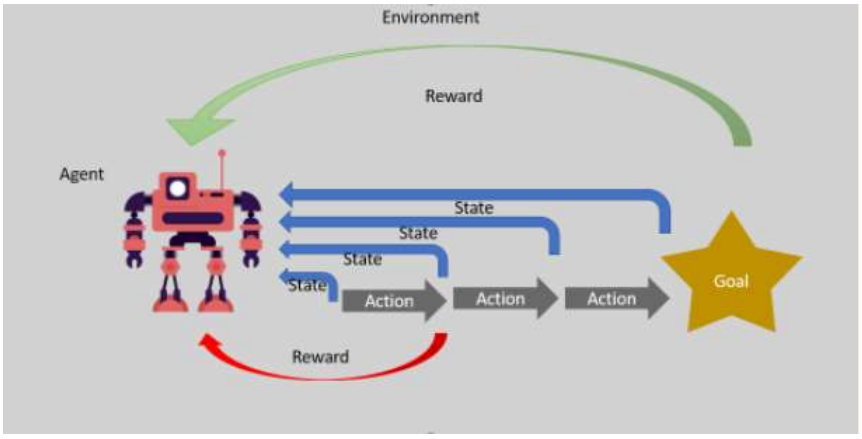


Gambar 2. 3 Cara Kerja *Unsupervised Learning*

(**Sumber:** *https://www.uc.ac.id/ict/perbedaan-supervised-learning-and-unsupervised-learning/*)

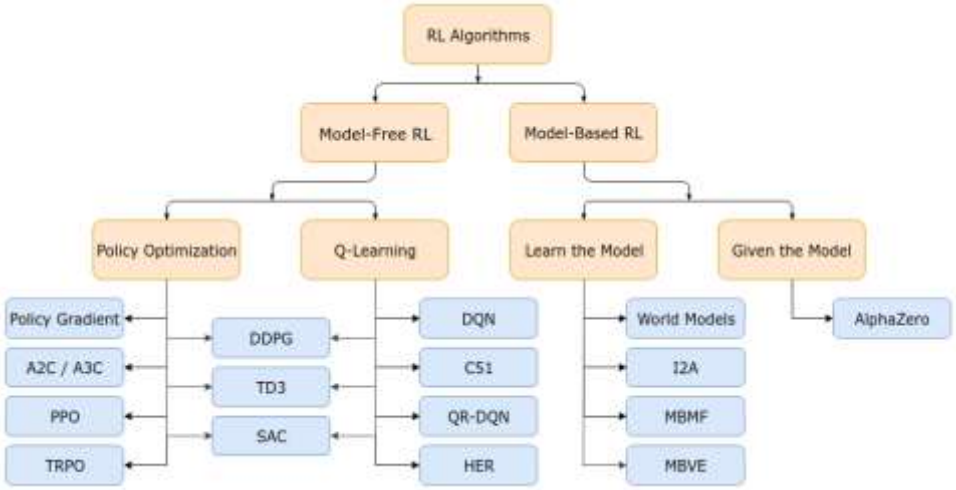
2.2.3 Reinforcement Learning

Reinforcement Learning (RL) adalah metodologi atau algoritma yang menerapkan prinsip yang dapat digunakan dengan jaringan saraf *(Neural Networks).* Inti dari *RL* adalah kebutuhan akan pelatihan. (*training*). Agen komputasi yang tidak memiliki pengetahuan kemudian *RL* memberinya data melalui beberapa proses otomatis untuk dipelajari (Lanham, Reinforcement Learning, 2019).



Gambar 2. 4 Cara Kerja *Reinforcement Learning*
(**Sumber:** Buku *Hands-On Deep Learning for Games*)

Diagram tersebut menunjukan bahwa *agent* (agen komputasi) akan mempelajari bagaimana memaksimalkan *reward (long-term reward)* yang diperoleh dari aksi *(action)* suatu kondisi *(state).* Perubahan aksi yang dilakukan agen dapat menghasilkan perubahan kondisi yang nantinya akan mempengarungi *reward* selanjutnya.



Gambar 2. 5 Algoritma *Reinforcement Learning*
(**Sumber:** *https://www.vpslabs.net/reinforcement-learning/#klasifikasi-machine-learning*)

Terdapat dua model pembelajaran penting dalam *Reinforcement Learning* yaitu:

a. *Model-Free Reinforcement Learning*

Model-Free Reinforcement Learning merupakan metode bebas model yang bergantung pada pembelajaran dalam konteks *Reinforcement Learning.* Misalnya model memprediksi keadaan *(state)* dan *reward* selanjutnya berdasarkan keadaan *(state)* dan tindakan *(action)* model. Lingkungan *Reinforcement Learning* dapat dijelaskan dengan proses keputusan *Markov Decision Process (MDP)* (Sagar, 2021).

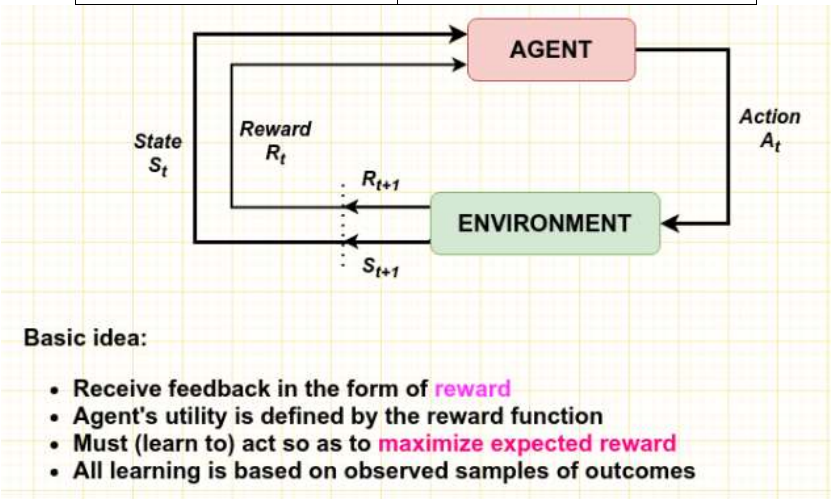
b. *Markov Decision Process (MDP)*

Markov Decision Process (MDP) merupakan pendekatan dimana keputusan yang diambil terdiri dari *state* dalam bentuk grid dengan membaginya menjadi *state, action, model/model transition* dan *reward. Policy* merupakan solusi untuk *MDP* untuk menemukan solusi yang optimal untuk tugas *MDP* tersebut. Parameter yang digunakan untuk solusi yang diharapkan adalah sebagai berikut (vpslabsTIM, 2020):

Tabel 2. 1 Parameter MDP

Istilah	Fungsi
Set of states	***S***
Set of actions	***A(s), A***
Transition	***T(s,a,s') ~ P(s'|s,a)***
Reward	***R(s), R(s,a), R(s,a,s')***

Policy	***Π***
Value	***V***



Gambar 2. 6 Cara kerja *Reinforcement Learning MDP*
(**Sumber:** *https://www.vpslabs.net/reinforcement-learning/#klasifikasi-machine-learning*)

Cara kerja *Reinforcement Learning (MDP)* yaitu mencoba berbagai pilihan dan kemungkinan yang berbeda dengan melakukan pengamatan *(observation)* dan evaluasi *(evaluation)* setiap pencapaian karena dapat belajar dari pengalaman.

Adapun klasifikasi dari *Model-Free Reinforcement Learning* yaitu:

1) *Policy Optimization*

   Policy Optimization merupakan optimalisasi pendekatan *Reinforcement Learning* yang efektif untuk menyelesaikan tugas kontrol yang dilakukan terus menerus dengan kebijakan linier dan mendalam (Metelli, 2018).

2) *Q-Learning*

Q-Learning merupakan algoritma Reinforcement Learning tanpa model *(model free algorithm)*untuk mempelajari *policy* dan mengajari agen komputasi tindakan apa yang harus diambil disesuaikan dengan keadaan. *Q-Learning* tidak memerlukan *brain model* dari *environment* dan menangani masalah dengan transisi stokastik dan *reward* tanpa melakukan adaptasi (vpslabsTIM, 2020).

3) *Model-Based Reinforcement Learning*

Model-Based Reinforcement Learning merupakan metode berbasis model yang bergantung pada perencanaan sebagai komponen utama dan mengacu pada pembelajaran perilaku optimal *(optimal behavior)* dengan mempelajari lingkungan *(environment)* dengan mengambil tindakan *(action)* dan mengamati hasil yang mencakup *state* dan *reward* selanjutnya (Sagar, 2021) (Soumya Ray, 2011).

Terdapat 2 klasifikasi *Model-Based Reinforcement Learning* yaitu:

1. *Learn the Model*

   Metode berbasis model pada *Reinforcement Learning* yang mengacu dengan mempelajari modelnya seperti *World Model, I2A, MBMF, MBVE*

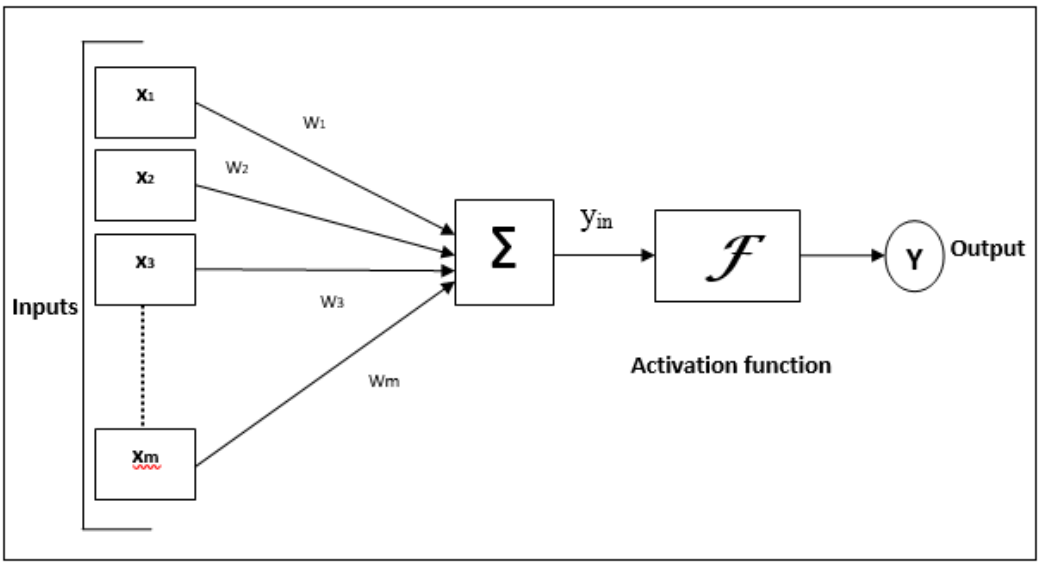
2. *Given the Model*

   Metode berbasis model pada *Reinforcement Learning* yang mengacu pada model yang disediakan / mengingat modelnya seperti *Alpa Zero*

2.3 Deep Learning

Deep Learning atau teknik pembelajaran mendalam merupakan bagian dari metode yang membentuk pembelajaran mesin, berdasarkan karakteristik data. Tingkat keberhasilan dalam memprediksi *class* dan klasifikasi sangat tinggi karena menggunakan jaringan saraf yang dibagi menjadi beberapa lapisan. Jaringan saraf adalah struktur yang dibagi menjadi satu set elemen yang saling berhubungan *(neuron)* yang memodelkan sistem saraf buatan. Struktur ini terinspirasi oleh jaringan saraf biologis yang membentuk otak manusia (Velasco, 2018). Konsep inti dari pembuatan jaringan saraf tiruan bukanlah hal baru, namun evolusi dari *hardware* modern menyediakan kekuatan komputasi yang cukup untuk melatih jaringan tersebut secara efektif contohnya *frameworks* seperti *TensorFlow, Keras* dan *Torch,* yang dapat membuat dan membangun model *Machine Learning* jauh lebih nyaman. (Osiński & Budek, 2018).

Gambar berikut menunjukan model umum *Artificial Neural Network (ANN)* diikuti dengan pemprosesannya (Tutorialspoint, 2021):



Gambar 2. 7 Representasi *Artificial Neural Network*

(**Sumber:** *https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_quick_guide.htm*)

Net input pada model jaringan saraf tiruan *ANN* yang umum, dapat dihitung sebagai berikut:

$$y_{in} = x_1.w_1 + x_2.w_2 + x_3.w_3 \ldots x_m.w_m$$

Yaitu, *net input* $y_{in} = \sum_i^m x_i.w_i$ Output dapat dihitung dengan menerapkan fungsi aktivasi *(activation function)* melalui *net input.*

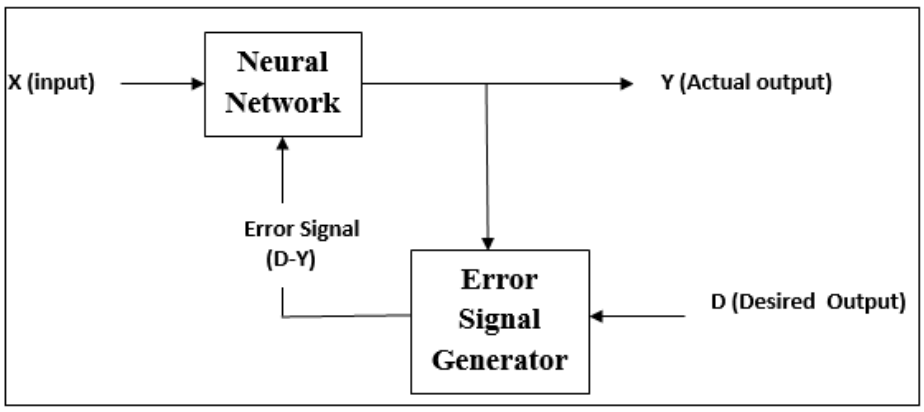
$$Y = F(y_{in})$$

Output = fungsi (kalkulasi *net input*)

Penyesuaian bobot atau pembelajaran dalam jaringan saraf tiruan *(ANN)* merupakan metode yang memodifikasi bobot koneksi antara *neuron* dari jaringan tertentu. Pembelajaran di *ANN* dapat diklasifikasikan ke dalam tiga kategori yaitu pembelajaran yang diawasi *(Supervised Learning),* pembelajaran tanpa pengawasan *(Unsupervised Learning),* dan pembelajaran penguatan *(Reinforcement Learning)* (Tutorialspoint, 2021).

a. *Supervised Learning*

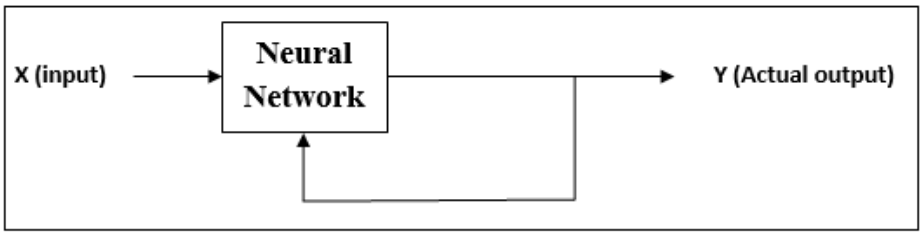
   Jenis pembelajaran ini bersifat mandiri atau tanpa pengawasan dimana *input vector* dari jenis yang sama digabungkan untuk membentuk *cluster.* Ketika pola input baru diterapkan, maka jaringan saraf memberikan respon *output* yang menunjukan *class* dimana pola *input* itu berasal. Jaringan itu sendiri harus menemukan pola dan fitur dari data *input* dan hubungan antara data *input* dengan *output*.



Gambar 2. 8 *Supervised Learning dalam Artificial Neural Network* (**Sumber:***https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_quick_guide.htm*)

b. *Unsupervised Learning*

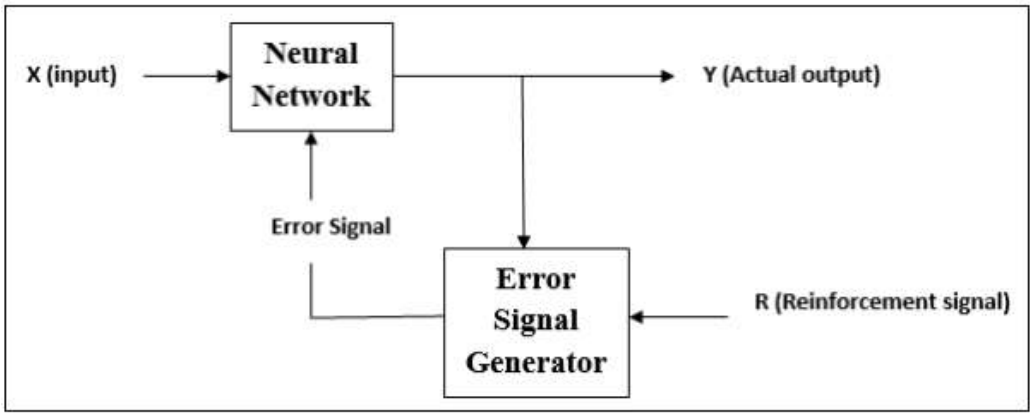
Jenis pembelajaran ini tergantung pada proses pembelajaran terawasi, dimana vektor *input* terhubung yang ke dalam jaringan akan memberikan *output* aktual, selanjutnya akan dibandingkan dengan vektor *output* yang diinginkan. Sinyal eror dihasilkan jika ada perbedaan antara output aktual dan vektor *output* yang diinginkan, maka bobot disesuaikan sampai *output* aktual sesuai dengan *output* yang diinginkan.



Gambar 2. 9 *Unsupervised Learning dalam Artificial Neural Network* (**Sumber:** *https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_quick_guide.htm*)

c. *Reinforcement Learning*

Jenis pembelajaran ini digunakan untuk memperkuat jaringan atas beberapa informasi kritis. Selama proses pelatihan, jaringan menerima bebrapa umpan balik dari lingkungan *(environment).* Pembelajaran ini mirip dengan *supervised learning* namun umpan balik yang diperoleh *reinforcement learning* bersifat evaluatif bukan instruktif. Setelah menerima umpan balik, jaringan melakukan penyesuaian bobot untuk mendapatkan informasi kritik yang lebih baik di masa depan.



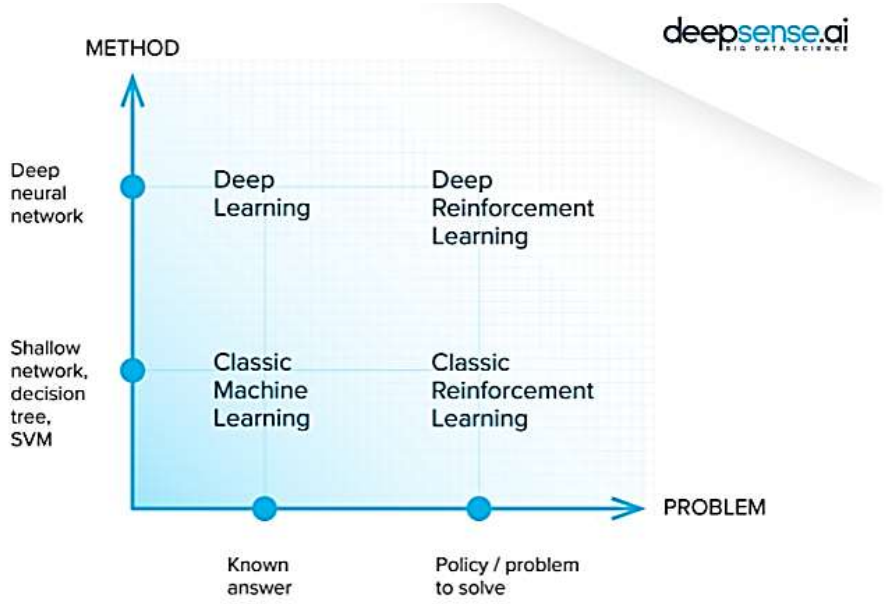
Gambar 2. 10 *Reinforcement Learning dalam Artificial Neural Network*

(**Sumber:** *https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_quick_guide.htm*)

2.4 Deep Reinforcement Learning

Deep Reinforcement Learning (DLR) adalah salah satu metodologi *Artificial Intelligence (AI)* terbaru yang telah diterapkan di beberapa bidang industri (Skevofylakas, Deep Reinforcement Learning using Unity ML-Agents — part I, 2020). Faktanya dalam beberapa riset mengatakan bahwa tidak ada pembagian yang jelas antara *machine learning, deep learning* dan *reinforcement learning.* Ini

merupakan relasi persegi dalam parallelogram, dimana *deep reinforcement learning* adalah sub bagian dari *machine learning*. Menggunakan cara yang sama *reinforcement learning* adalah aplikasi khusus *machine* dan *deep learning* yang merancang untuk memecahkan masalah dengan cara tertentu (Osiński & Budek, 2018).



Gambar 2. 11 Relasi *Machine Learning dan Deep Reinforcement Learning* (**Sumber:** *https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide*)

2.4.1 Definisi *Deep Reinforcement Learning*

DLR merujuk pada pendekatan pembelajaran mesin di mana agen belajar untuk mengambil keputusan tindakan dalam lingkungan tertentu untuk memaksimalkan *reward* yang diperoleh. Metode ini mengandalkan konsep *trial-and-error*, di mana agen belajar dari pengalaman saat berinteraksi dengan lingkungan untuk mencapai tujuan tertentu (Arulkumaran, Deisenroth, Brundage, & Bharath, 2017).

2.4.2 Konsep *Deep Reinforcement Learning*

a. Reinforcement Learning (RL)

Dasar dari *DLR* adalah konsep *RL* yang melibatkan agen yang melakukan tindakan di lingkungan tertentu untuk mencapai tujuan tertentu dan mendapatkan *reward* (Wei, 2023).

b. Deep Neural Networks (DNN)

Dalam *DLR*, jaringan saraf yang dalam digunakan untuk merepresentasikan nilai-nilai *state* dan *action* yang kompleks, yang memungkinkan agen untuk mempelajari strategi yang lebih abstrak dan adaptif (Alzubaidi, et al., 2021).

c. Penggabungan *RL* dengan *DNN*:

DLR menggabungkan kekuatan *RL* dalam pembelajaran tindakan dari *reward* dengan kemampuan representasi kompleks dari *DNN*, memungkinkan agen untuk memahami dan belajar dari lingkungan yang kompleks.

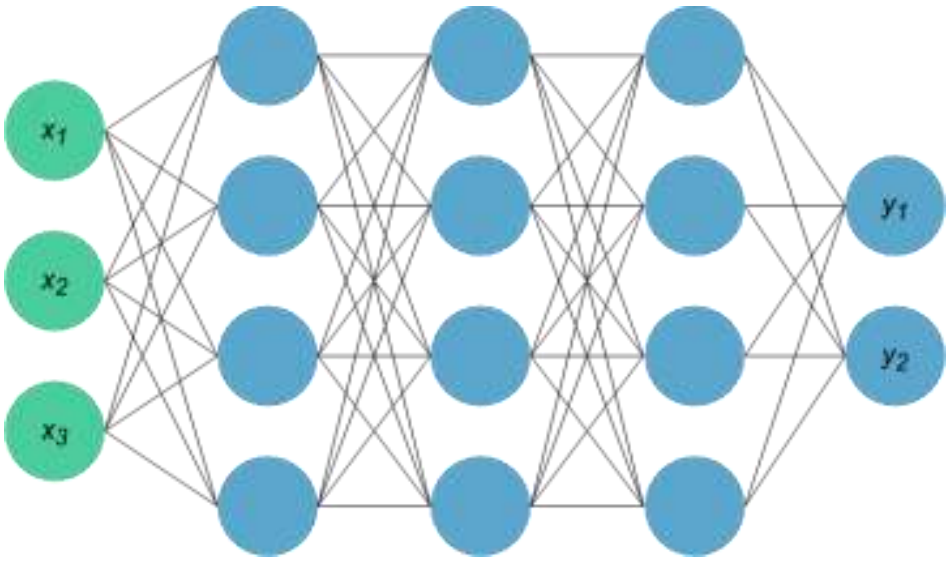
2.4.3 Keterkaitan dengan Pembelajaran Penguatan

Pembelajaran Penguatan (*Reinforcement Learning*) adalah dasar dari *DLR*, yang merupakan pengembangan dari *RL* dengan memanfaatkan konsep jaringan saraf yang dalam untuk meningkatkan kemampuan belajar agen dalam lingkungan yang lebih kompleks.

DLR memanfaatkan prinsip-prinsip *RL*, seperti *state, action*, dan *reward*, namun dengan representasi yang lebih kompleks dan adaptif melalui jaringan saraf yang dalam untuk mempelajari keputusan yang lebih abstrak.

2.4.4 Arsitektur Jaringan Saraf Tiruan yang Dalam pada *DLR*

Arsitektur jaringan saraf tiruan yang dalam (*Deep Neural Network - DNN*) dalam *Deep Reinforcement Learning (DLR)* menjadi kunci dalam mengembangkan representasi kompleks dari lingkungan permainan atau masalah yang dihadapi oleh agen.



Gambar 2. 12 *Deep Neural Network*

(**Sumber:** *https://structilmy.com/blog/2019/09/02/sekilas-tentang-jaringan-saraf-tiruan-dan-deep-learning/*)

Beberapa arsitektur yang sering digunakan dalam *DLR* termasuk:

a. Convolutional Neural Networks (CNNs)

1) **Deskripsi**: Biasanya digunakan dalam pengolahan gambar untuk mempelajari fitur-fitur visual yang kompleks dari *input* pixel.
2) **Penerapan dalam *DLR***: Digunakan untuk memproses citra atau input visual dari permainan, memungkinkan agen untuk mengidentifikasi pola dan fitur penting dalam lingkungan permainan seperti *Flappy Bird*.

b. Deep Q-Networks (DQN)

1) **Deskripsi**: Merupakan arsitektur yang digunakan dalam kombinasi dengan metode *Q-learning*. *DQN* memanfaatkan beberapa lapisan konvolusi dan lapisan-lapisan terhubung secara penuh.
2) **Penerapan dalam *DLR***: *DQN* telah digunakan dalam mempelajari keputusan tindakan dari nilai-nilai *Q* yang diestimasi dengan menggunakan jaringan saraf yang dalam.

c. Recurrent Neural Networks (RNNs)

1) **Deskripsi**: Jenis jaringan saraf yang mempertahankan hubungan temporal dan memori dalam urutan data input.
2) **Penerapan dalam *DLR***: Digunakan dalam skenario di mana urutan tindakan atau informasi yang terkait dengan waktu penting, seperti dalam pengendalian robotik atau permainan dengan alur waktu yang berkesinambungan.

d. *Actor-Critic Networks*

1) **Deskripsi**: Merupakan gabungan dari dua arsitektur: "*actor*" yang menghasilkan tindakan berdasarkan keadaan lingkungan dan "*critic*" yang menilai keputusan tersebut.
2) **Penerapan dalam *DLR***: Digunakan untuk mempelajari kebijakan (*policy*) dan mengevaluasi nilai (*value*) tindakan agen dalam lingkungan yang dinamis.

e. *Dueling Neural Networks*

1) **Deskripsi**: Merupakan arsitektur yang memisahkan representasi nilai keadaan (*state values*) dan nilai-kelebihan (*advantage values*) untuk mempercepat dan memperbaiki pembelajaran (Wang, et al., 2016).

2) **Penerapan dalam *DLR***: Digunakan untuk meningkatkan kecepatan konvergensi dalam mempelajari nilai dari tindakan dan keadaan lingkungan.

2.4.5 Keunggulan *DLR* dalam Lingkungan Permainan:

a. Kemampuan Adaptasi yang Tinggi

Deep Reinforcement Learning (DLR) mampu belajar dari pengalaman interaksi dengan lingkungan, memungkinkan agen untuk beradaptasi secara efektif terhadap perubahan yang terjadi. Melalui *proses trial* and *error*, agen *DLR* terus memperbarui strateginya, menjadikannya lebih tangguh dalam menghadapi kondisi dinamis dan situasi yang kompleks.

b. Pembelajaran dari Data Tak Terstruktur

Deep Reinforcement Learning (DLR) mampu bekerja dengan data tidak terstruktur, seperti citra visual dalam permainan, tanpa memerlukan fitur yang telah terdefinisi. Dengan jaringan saraf dalam, *DLR* mengekstrak pola langsung dari data mentah dan belajar dari pengalaman. Pendekatan ini memungkinkannya mengoptimalkan keputusan dalam situasi kompleks, seperti dalam permainan atau robotika.

c. Kemampuan Mempelajari Strategi yang Kompleks

DLR memungkinkan agen untuk mempelajari strategi yang sangat kompleks dan adaptif dalam lingkungan yang dinamis, seperti permainan yang memiliki banyak kemungkinan aksi.

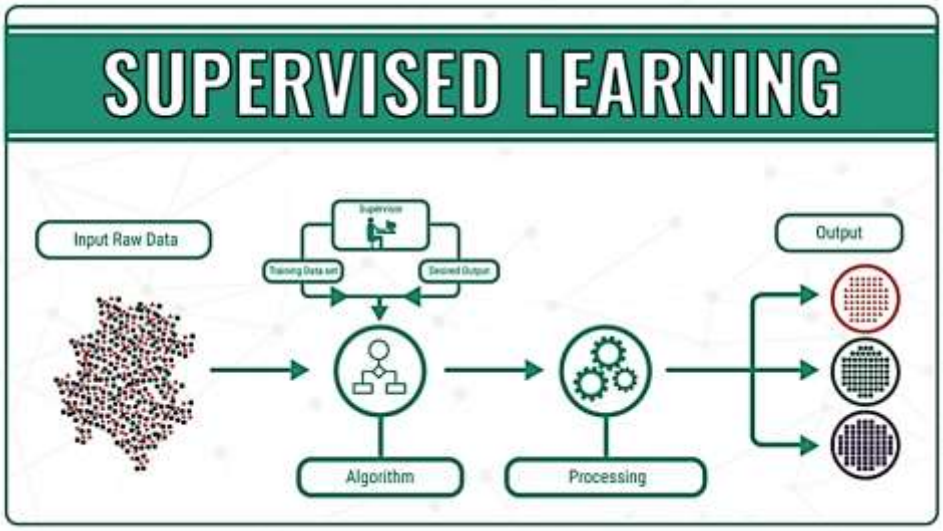
d. Penerapan pada Berbagai Macam Permainan

DLR dapat diterapkan pada berbagai jenis permainan mulai dari permainan papan hingga permainan *video* yang lebih kompleks, seperti strategi *real-time* dan *game-platform*.

2.5 Supervised Learning

2.5.1 Definisi *Supervised Learning*

Supervised Learning adalah salah satu paradigma dalam pembelajaran mesin di mana algoritma belajar dari data yang telah dilabeli sebelumnya. Dalam metode ini, model atau algoritma belajar untuk membuat prediksi atau mengidentifikasi pola dari data yang telah diketahui labelnya. Tujuannya adalah untuk membuat model yang mampu memetakan *input* ke *output* yang tepat berdasarkan contoh-contoh yang telah diberikan (Wang, Lin, & Dang, 2020).



Gambar 2. 13 *Supervised Learning*

(**Sumber:** *https://www.techfor.id/konsep-supervised-learning-dalam-membangun-model-machine-learning/*)

2.5.2 Konsep Supervised *Learning*

a. Data Terlabel

Supervised Learning menggunakan data yang sudah memiliki label atau klasifikasi yang jelas.

b. Pengenalan Pola

Algoritma mencari pola dalam data yang terlabel untuk dapat memprediksi atau mengklasifikasikan data baru.

c. Fungsi Objektif yang Diketahui

Tujuan dari model yang dibuat dalam *Supervised Learning* sudah diketahui, seperti minimisasi *error* prediksi atau klasifikasi yang salah.

2.5.3 Metode Supervised *Learning*

Supervised Learning mencakup beberapa metode, di antaranya:

a. Regresi

Prediksi nilai kontinu dari data, misalnya memprediksi harga rumah berdasarkan fitur-fiturnya.

b. Klasifikasi

Klasifikasi data menjadi kategori atau kelas tertentu, seperti mengidentifikasi apakah sebuah *email* masuk merupakan *spam* atau bukan.

c. Deteksi Objek

Identifikasi objek dalam gambar atau *video*, seperti pengenalan wajah atau deteksi kendaraan.

d. Pengenalan Pola

Identifikasi pola yang mendasari data, sering digunakan dalam aplikasi medis untuk diagnosis atau dalam keuangan untuk prediksi tren.

2.5.4 Kelebihan *Supervised Learning* dalam Konteks Permainan

a. Prediksi yang Akurat

Supervised Learning, ketika dilatih dengan data yang tepat, dapat memberikan prediksi atau pengambilan keputusan yang akurat berdasarkan contoh-contoh yang telah dilabeli sebelumnya.

b. Penggunaan Label untuk Pembelajaran

Dengan label yang jelas pada data pelatihan, *Supervised Learning* bisa menghasilkan model yang mampu memahami pola dan strategi yang dibutuhkan dalam permainan, seperti pemilihan aksi yang optimal pada situasi tertentu.

c. Kemampuan Mengidentifikasi Pola yang Kompleks

Supervised Learning mampu mempelajari pola-pola yang kompleks dari data pelatihan, yang dapat berguna dalam mengenali pola-pola permainan yang memerlukan respons atau tindakan yang tepat.

2.5.5 Keterbatasan *Supervised Learning* dalam Konteks Permainan

a. Ketergantungan pada Data Terlabel

Dalam permainan yang kompleks, mendapatkan data terlabel yang cukup dan representatif bisa menjadi sulit dan mahal. Pengumpulan data yang berkualitas bisa menjadi tantangan.

b. Keterbatasan Generalisasi

Model *Supervised Learning* bisa cenderung *overfitting*, yaitu terlalu fokus pada data pelatihan tertentu dan gagal menggeneralisasi pola-pola yang ditemukan ke situasi yang belum pernah ditemui sebelumnya.

c. Ketidakpastian dalam Tindakan Tepat

Dalam beberapa kasus, prediksi yang akurat dari *Supervised Learning* mungkin tidak cukup untuk menentukan tindakan yang optimal dalam permainan yang dinamis dengan banyak kemungkinan.

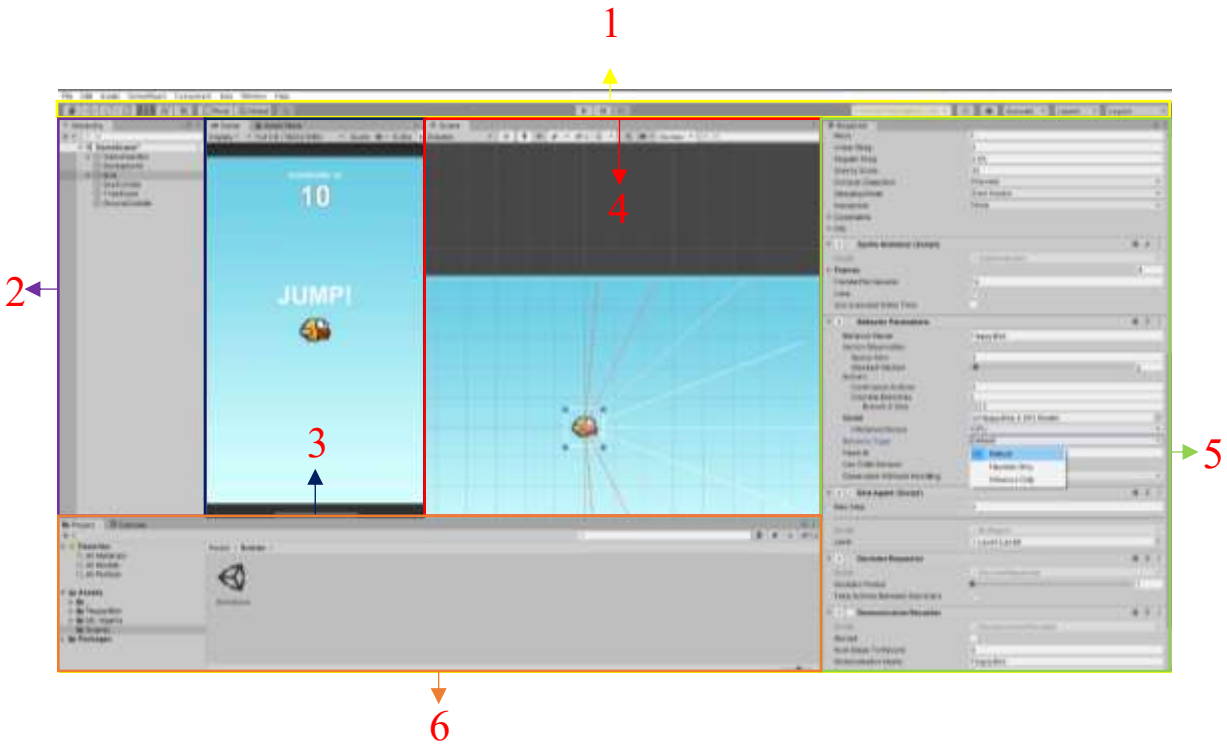
d. Kesulitan dalam Menangani Perubahan Lingkungan

Permainan seringkali melibatkan perubahan-perubahan yang dinamis dalam lingkungan. Model *Supervised Learning* mungkin kesulitan dalam menyesuaikan diri terhadap perubahan-perubahan ini tanpa data yang sesuai.

2.6 Unity

Unity merupakan *software engine* untuk mengembangkan *game casual*, *Augmented Reality (AR),* dan *Virtual Reality (VR)* yang dapat menambahkan *script* untuk memprogramnya (Azqiya, 2021).

2.6.1 User Interface Unity

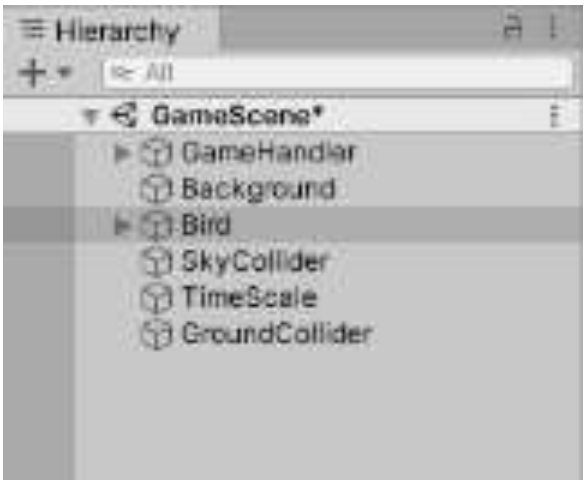


Gambar 2. 14 *User Interface Unity*

a. *Toolbar* berfungsi untuk menyediakan tombol tool yang mengatur berbagai *GameObject* atau komponen dalam *game scene.*

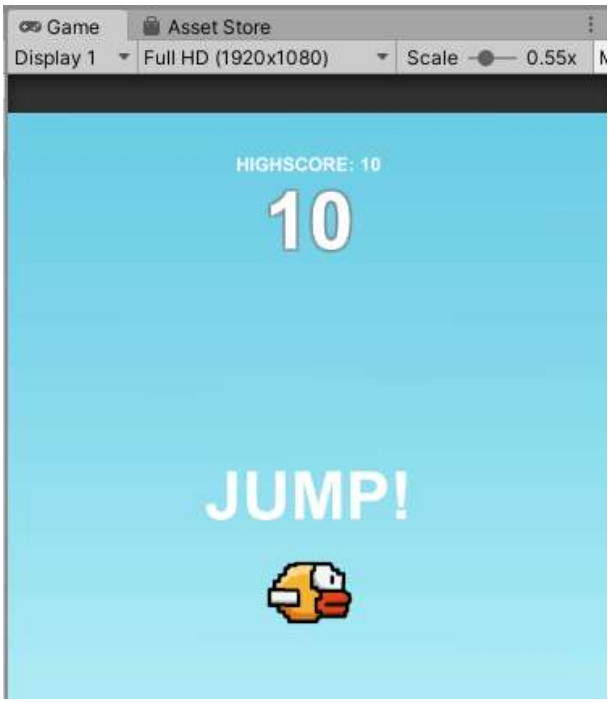
Gambar 2. 15 *Toolbar*

b. *Hierarchy* merupakan sebuah window yang berisi kumpulan *GameScene, GameObject*, seperti *3D Object, light, UI, Sound, Effect, Video* dan *Camera*.



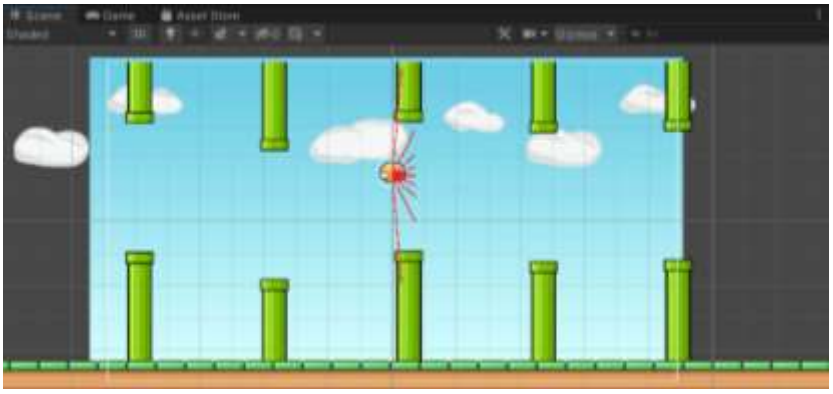
Gambar 2. 16 *Hierarchy*

c. *Game View* merupakan sebuah window yang menampilkan permainan yang sedang dijalankan. Ketika tombol *Play* di klik maka simulasi dimulai.



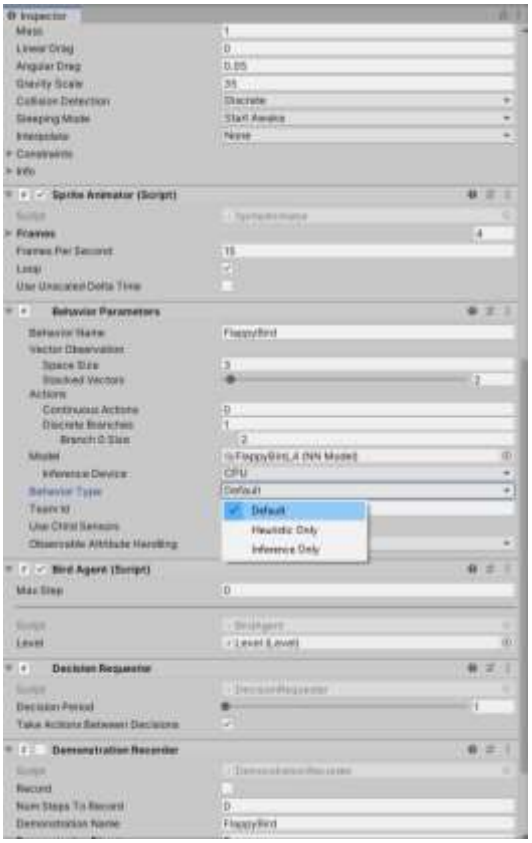
Gambar 2. 17 *GameView*

d. *Scene View* berfungsi untuk mengatur dan membuat tampilan *game*.



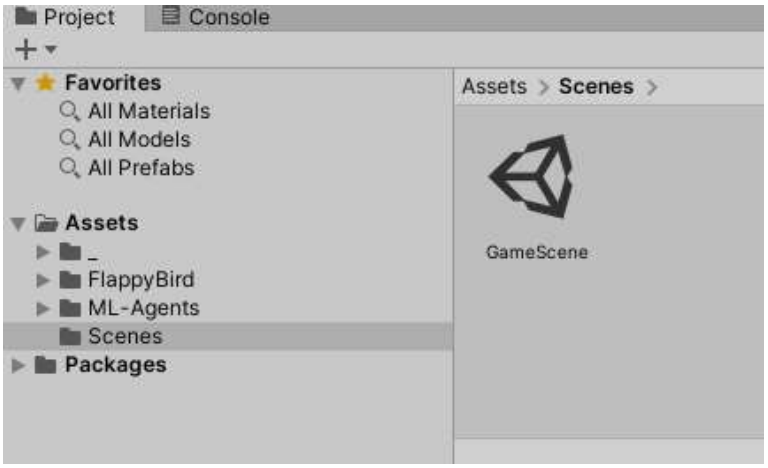
Gambar 2. 18 *Scene View*

e. *Inspector* berfungsi untuk melihat dan mengedit nilai dari properti *GameObject* yang dipilih saat ini.



Gambar 2. 19 *Inspector*

f. *Project Window* berfungsi untuk menyimpan semua *asset* yang digunakan untuk membuat sebuah *game* seperti *script, texture, 3D model, audio clip, image, material, animation* dll yang disimpan dalam *harddisk* komputer.



Gambar 2. 20 *Project Window*

2.6.2 Machine Learning Agents

Machine Learning Agents (*ML-Agents*) adalah proyek *open-source* dari *Unity* berupa *toolkit* yang memungkinkan *game* dan simulasi berfungsi sebagai *environment* (lingkungan) untuk melatih kecerdasan agen yang disimulasikan menggunakan *Unity Editor* dan berinteraksi melalui *API Python* (tsibiski, 2021) (Arthur Juliani, 2018). Toolkit menyediakan *ML-Agents SDK* yang berisi semua fungsionalitas untuk mendefinisikan *environment* dalam *Unity Editor* bersama dengan skrip inti C#.

Add-on ini dibuat sebagai cara untuk menciptakan lingkungan yang kompleks dan mempelajari berbagai jenis algoritma yang memungkinkan pengembang *game* dapat menerapkannya pada agen untuk mencapai berbagai tujuan (A. Juliani, 2018).

Machine Learning dan Deep Reinforcement Learning (DLR) telah diterapkan pada aplikasi *Unity* untuk menghasilkan *Deep Reinforcement Learning SDK* yang berfungsi sebagai pengembang di dalam *game* dan simulasi. Sebagai dasar untuk mesin *DLR*-nya *Unity* menggunakan model *Proximal Policy Optimization (PPO)* yang mudah digunakan dan menyediakan beberapa alat/tools untuk membantu mempelajari konsep kedepannya (Lanham, Unity ML-Agents, 2019).

2.6.3 Tensorflow

Tensorflow digunakan sebagai alat untuk memberikan kecerdasan pada agen komputasi dalam *Unity.* Algoritma yang digunakan dengan *Tensorflow* menggunakan *Deep Learning* sebagai teknik pembelajarannya. *Tensorflow* ini menyediakan fungsionalitas seperti melatih jaringan saraf, menguraikan pola berdasarkan algoritma untuk *machine learning* (Velasco, 2018).

Unity memiliki *Tensorflow* yang dilengkapi dengan seperangkat alat grafik yang disebut *TensorBoard.* Kegunaan *TensorBoard* yaitu untuk memantau kemajuan selama masa *training* yang dapat memvisualkan proses pembelajaran yang berguna untuk melihat performa agen yang dapat ditingkatkan (Lanham, Unity ML-Agents, 2019).

2.6.4 Flappy Bird

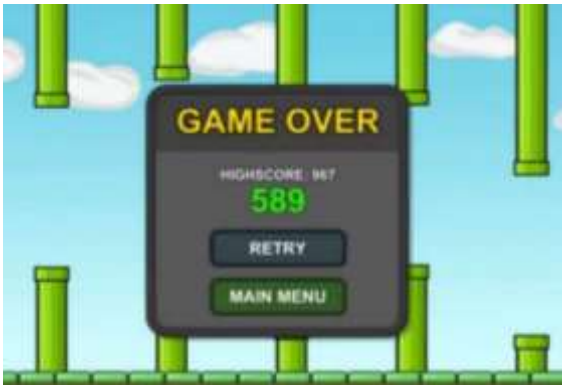
Flappy Bird merupakan sebuah *game* seluler yang bisa diakses dan diunduh secara gratis melalui *Appstore* dan *IOS. Flappy Bird* diluncurkan Mei 2013 oleh pengembang asal Vietnam yang bernama Dong Nguyen dan diterbitkan oleh *DotGEARS STUDIO* (Artyom, 2020).

Dikritik karena desain dan sifat adiktifnya mirip dengan *Super Mario,* maka pada Februari 2014 *Flappy Bird* dihapus dari *Appstore* dan *IOS.* Namun banyak pengembang lain yang membuat *cloning Flappy Bird* untuk memenuhi hasrat para pemain *game Flappy Bird* berbasis android maupun dekstop. Adapun alasan dari para pengembang lainnya untuk mengembangkan kembali *game Flappy Bird* guna memberikan inspirasi terkait penelitian yang dilakukan oleh para peneliti di bidang *Artificial Intelligence.*

Karakter utama dalam permainan *Flappy Bird* adalah seekor burung. Cara memainkannya cukup dengan mengetuk jari ke layar untuk membuat burung terbang/melompat dan burung itu harus melewati rintangan berupa pipa berwarna hijau (Anjungroso, 2014).

Gambar 2. 21 *Scene Flappy Bird*

Skor yang diterima pemain saat melewati pipa bernilai 1 dan jumlah skor pada *game* ini tidak terbatas. Jika saat mengklik *mouse button* / mengetuk jari pada *touchpad* tidak tepat atau bahkan menabrak pipa, maka burung akan jatuh dan skor terakhir muncul sebagai pertanda permainan kalah atau berakhir.



Gambar 2. 22 *Notifikasi saat permainan berakhir*

Permainan akan dimulai kembali dengan mengklik *Retry button* pada notifikasi yang muncul saat permainan berakhir *(Game Over)*. *Highscore* muncul setelah ada riwayat permainan yang dibandingkan dengan skor baru.

Adapun konsep pada penelitian ini merupakan adaptasi dari *game flappy bird* yang menggunakan peraturan permainan yang sama, namun cara memainkannya dimainkan secara otomatis oleh sistem yang dibuat dalam otak agen. Algoritma yang dipakai dalam penelitian ini mampu melatih agen untuk dapat berfikir dengan mengobservasi lingkungan, memutuskan tindakan secara mandiri untuk mendapatkan *reward* yang tinggi dengan kemampuan *superhuman*.

2.7 Penelitian *Terdahulu*

Penelitian terdahulu merupakan salah satu referensi dalam melakukan sebuah penelitian, tujuannya agar dapat memperkaya teori yang digunakan dalam tinjauan penelitian. Berdasarkan penelusuran sebelumnya, tidak ditemukan judul penelitian dengan nama yang sama dengan penelitian ini. Adapun pengutipan dari beberapa penelitian yang digunakan sebagai referensi untuk memperkaya literatur penelitian adalah sebagai berikut:

Tabel 2. 2 Penelitian Terdahulu

No	Nama Penulis	Judul	Hasil
1.	**Mario Rubak (2021)**	***Imitation Learning with the Unity Machine Learning Agents Toolkit (A feasibility study to create artificial players)***	**Penelitian ini mengembangkan jenis *NPC (Non-player character)* yang dikendalikan oleh *AI* untuk memainkan *game* sebagai pengganti manusia yang dapat disebut *Artificial Player*. Umumnya pengembangan *game Artificial Player* dikembangkan dengan menulis skrip untuk memberi**

No	Nama Penulis	Judul	Hasil
			pengetahuan tentang apa yang harus dilakukan dan dapat bertindak dengan cerdas. **Metode dalam penelitian ini menerapkan pendekatan *Reinforcement Learning* yang memungkinkan *Artificial Player* dapat belajar sendiri cara memainkan permainan. Permainan yang lebih kompleks membutuhkan banyak waktu pelatihan (*training*), maka solusi yang diterapkan yaitu dengan menambahkan *Imitation Learning* guna menambahkan kecerdasan buatan yang diterapkan pada *Artificial Player*.** **Kontribusi dari penelitian ini adalah untuk menunjukan bagaimana *Imitation Learning* dapat membantu melatih agen**

No	Nama Penulis	Judul	Hasil
			melalui *Unity Machine Learning Agents Toolkit (ML-Agents)* dalam upaya mengatasi kekurangan yang hanya menggunakan metode *Reinforcement Learning*. **Hasil penelitian menunjukan bahwa penerapan *Imitation Learning* dengan *ML-Agents* dapat meningkatkan kemampuan agen untuk memahami tujuan permainan lebih dari hanya menggunakan *extrinsic reward*, tetapi masih dibatasi oleh kompleksitas dan mekanisme permainan.**
2.	**Tokey Tahmid, Mohammad Abu Lobabah, Muntasir Ahsan, Raisa Zarin,**	***Character Animation Using Reinforcement Learning and Imitation Learning Algorithms***	**Penelitian ini mengusulkan model yang bertujuan untuk menghasilkan *real-time* pada karakter animasi sebagai *biped locomotion* dalam *Unity ML-Agents* dengan menggunakan *Algoritma Reinforcement Learning* dan *Imitation Learning*.**

No	Nama Penulis	Judul	Hasil
	Sabah Shahnoor Anis (2021)		**Hasil evaluasi pelatihan yang dilakukan hanya dengan bawaan algoritma *Reinforcement Learning* yaitu *Proximal Policy Optimization (PPO)* dengan mengkombinasikan algoritma *Imitation Learning* pada *Behavior Cloning & Generative Adversarial Imitation Learning (BC & GAIL).*** **Perbandingan antara dua set data pelatihan menunjukan bahwa model mampu menghasilkan animasi secara *real-time* dengan menghindari *database* yang besar, demonstrasi pada penelitian ini menghasilkan jumlah kompresi data yang baik sehingga mudah dilakukan dengan tetap menjaga kualitasnya.**

No	Nama Penulis	Judul	Hasil
3.	**Johan Fröberg & Carl Håkansson (2021)**	***Application of machine learning to construct advanced NPC behaviors in Unity 3D***	**Tujuan proyek pada penelitian ini telah tercapai. Eksperimen dan pengujian dalam tesis ini menunjukkan bahwa sangat mungkin untuk membuat agen yang memandu pemain melalui jalur yang telah ditentukan saat memainkan level. Ternyata ini dimungkinkan dengan berbagai jenis algoritme pembelajaran mesin untuk lingkungan yang relatif sederhana, karena hasil yang sukses dengan *PPO* dan campuran antara *Imitation Learning* dan *MA-POCA*. Perilaku seperti manusia dapat diciptakan untuk agen dengan beberapa hadiah sederhana, seperti hadiah ketika pemain melewati area target dan hukuman ketika melewati area yang salah. Namun, tidak semua algoritma *ML* bekerja sama**

No	Nama Penulis	Judul	Hasil
			baiknya di setiap situasi. Penting untuk terlebih dahulu mempertimbangkan perilaku apa yang Anda inginkan dari agen sebelum memilih metode. Pembelajaran imitasi, misalnya, tidak cocok jika Anda ingin membuat sekelompok agen yang berkolaborasi, dan pembelajaran penguatan tidak cocok di lingkungan kompleks yang besar karena menjadi sulit untuk membuat struktur penghargaan yang sesuai. **Menciptakan perilaku *NPC* yang menunjukkan ciri-ciri yang sangat spesifik agar sesuai dengan *gameplay* atau narasi dapat dilakukan dengan menggunakan pembelajaran mesin dengan hati-hati membangun lingkungan belajar dan struktur penghargaan yang mendorong perilaku ini. Menggunakan**

No	Nama Penulis	Judul	Hasil
			pembelajaran imitasi juga dapat menjadi metode yang efektif, karena gaya bermain yang diinginkan dapat direkam dan diberikan ke algoritma pembelajaran. Namun sulit untuk menghasilkan perilaku yang diinginkan secara andal, karena struktur penghargaan yang berbelit-belit dapat menghasilkan perilaku yang muncul yang tidak diinginkan. **Ada beberapa detail menarik yang dapat dikerjakan lebih lanjut dengan proyek ini. Untuk menjadikannya *game* zombie, detail penting adalah membuat pembelajaran berbasis kelompok berjalan dengan baik. Peluang terbaik untuk mendapatkan simulasi kerja dengan sekelompok agen adalah terus bereksperimen dengan algoritme *MA-POCA***

No	Nama Penulis	Judul	Hasil
			bawaan *Unity*, karena telah terbukti berfungsi untuk lingkungan multiagen kooperatif di masa lalu. Dengan lebih banyak langkah, ada kemungkinan bahwa penghargaan kumulatif untuk grup dapat menyatu menuju hasil yang baik, tetapi beberapa perubahan dalam struktur penghargaan mungkin juga diperlukan. **Ketika agen telah menjadi cukup pintar dan pembelajaran berbasis kelompok berhasil, maka dimungkinkan untuk lebih fokus pada *gameplay* itu sendiri, menyeimbangkan permainan dan grafik**

No	Nama Penulis	Judul	Hasil
4.	**Bakhtiyar Ospanov (2021)**	***Training intelligent tennis adversaries using self-play with ML agents***	**Penelitian ini mengusulkan lingkungan simulasi dengan konteks visual dan fisik yang kaya untuk melatih agen cerdas untuk permainan tenis. Lingkungan dibuat dengan *Unity Real-Time Development Platform* yang memungkinkan simulasi yang tepat dari interaksi fisik, representasi grafis, dan perilaku logis dari objek dunia nyata. Lingkungan yang mapan memfasilitasi pengumpulan pengamatan yang akurat dan serbaguna selama pelatihan. Data ini dikirimkan ke lokasi pelatihan menggunakan jembatan oleh *ML-Agents Toolkit.* Paket ini menyediakan alat untuk melakukan pelatihan menggunakan pembelajaran penguatan dan merumuskan kebijakan. Ada dua model memuaskan yang**

No	Nama Penulis	Judul	Hasil
			dihasilkan sebagai output dari penelitian ini. Yang pertama digunakan dalam produksi seperti yang dijelaskan di atas dan yang kedua masih menjadi subjek penelitian. Model terakhir adalah work-in-progress karena batasan yang dipaksakan oleh lingkungan dan algoritma yang dipilih. Lingkungan tenis membutuhkan interaksi kompleks dari agen ML, termasuk kemampuan untuk memukul bola dari titik mana pun di dalam area lapangan. Ini menyiratkan kebebasan penuh dalam gerakan dan rotasi di sepanjang ketiga sumbu. Kebijakan terakhir dilatih dengan properti yang tidak terkunci ini tetapi dengan lingkungan yang dimodifikasi seperti yang dijelaskan sebelumnya. Jika tidak, algoritme

No	Nama Penulis	Judul	Hasil
			tidak dapat menangani ruang pencarian yang begitu besar dengan sinyal hadiah yang jarang. Kombinasi penguatan dan pembelajaran kurikulum tidak cukup bagi agen untuk mempelajari interaksi yang kompleks dengan kondisi yang diperpanjang. Salah satu cara perbaikan yang mungkin adalah dengan memperkenalkan pembelajaran imitasi ke dalam kombinasi ini. Ini membutuhkan demonstrasi yang direkam dari pengguna ahli 35 yang dapat dilakukan dengan lingkungan dan toolkit yang ada. Jika rangkaian demonstrasi yang dihasilkan berisi sejumlah kemungkinan keadaan yang terbatas, disarankan untuk menggunakan Pembelajaran Imitasi Permusuhan Generatif, jika tidak, Kloning Perilaku

No	Nama Penulis	Judul	Hasil
			dapat digunakan. Dalam lingkungan yang kompleks dengan imbalan yang jarang, juga disarankan untuk memperkenalkan penghargaan intrinsik, seperti sinyal penghargaan Curiosity dan distilasi Jaringan Acak. Semua pendekatan ini dapat digunakan secara terpisah maupun terintegrasi secara bersamaan. Ada arah potensial lain di mana pengaturan saat ini dapat disesuaikan untuk eksperimen baru. Versi ganda (berpasangan) **tenis dapat dilatih. Ini berarti memperkenalkan dua agen ML lagi dan memperluas area pengadilan. Model yang dilatih saat ini akan gagal menangani skenario ini karena tabrakan dua agen dari tim yang sama kemungkinan besar akan terjadi.**

No	Nama Penulis	Judul	Hasil
			Meskipun pengaturan dan alat lingkungan yang diberikan sudah cukup untuk bereksperimen dengan ide ini. Secara keseluruhan, tanpa mempertimbangkan kemungkinan peningkatan untuk batasan yang disajikan, *Unity* dapat digunakan sebagai platform simulasi umum untuk tujuan penelitian. Ini telah menghasilkan pelatihan agen tenis cerdas yang menunjukkan perilaku memuaskan dan digunakan sebagai lawan yang tidak dapat dimainkan dalam aplikasi pengujian.
5.	**Haven Brown (2020)**	***Applying Imitation and Reinforcement Learning to Sparse Reward Environments***	**Fokus dari penelitian ini adalah untuk mempersingkat waktu yang diperlukan untuk melatih agen pembelajaran penguatan agar berkinerja lebih baik daripada manusia**

No	Nama Penulis	Judul	Hasil
			dalam lingkungan hadiah yang jarang. Menemukan solusi tujuan umum untuk masalah ini sangat penting untuk menciptakan agen di masa depan yang mampu mengelola sistem besar atau melakukan serangkaian tugas sebelum menerima umpan balik. **Tujuan dari proyek ini adalah untuk membuat fungsi transisi antara algoritma pembelajaran imitasi (juga disebut sebagai algoritma kloning perilaku) dan algoritma pembelajaran penguatan. Tujuan dari pendekatan ini adalah untuk memungkinkan agen untuk terlebih dahulu belajar melakukan tugas dengan meniru tindakan manusia melalui algoritma pembelajaran imitasi dan kemudian belajar**

No	Nama Penulis	Judul	Hasil
			melakukan tugas lebih baik atau lebih cepat daripada manusia dengan pelatihan dengan algoritma pembelajaran penguatan. Proyek ini menggunakan *Unity3D* untuk memodelkan lingkungan hadiah yang jarang dan memungkinkan penggunaan *Mlagents toolkit* yang disediakan oleh *Unity3D*. Toolkit yang disediakan oleh *Unity3D* adalah proyek open source yang tidak memelihara dokumentasi untuk versi perangkat lunak sebelumnya, jadi perubahan besar baru-baru ini pada penggunaan alat *mlagents* dalam kode menyebabkan keterlambatan pencapaian tujuan yang lebih besar dari proyek ini. Oleh karena itu, makalah ini menguraikan pendekatan teoritis terhadap masalah tersebut

No	Nama Penulis	Judul	Hasil
			dan beberapa implementasinya di *Unity3D*. Ini akan memberikan gambaran menyeluruh tentang alat umum yang digunakan untuk melatih agen di lingkungan hadiah yang jarang, khususnya di lingkungan video *game*.
6.	**Amira E.Youssef, Sohaila El Missiry, Islam Nabil El-gaafary, Jailan S. ElMosalami, Khaled M. Awad, Khaled Yasser (2019)**	***Building your kingdom Imitation Learning for a Custom Gameplay Using Unity ML-agents***	**Penelitian ini menerapkan Imitation Learning untuk custom *game*play menggunakan *Unity ML-Agents*, dimana metode yang diterapkan adalah menciptakan perilaku (seperti manusia) dengan merekam action sendiri. Mendapatkan pengamatan dan tindakan itu sendiri didapat dari model pelatihan dari Reinforcement Learning.** **Imitation Learning pada Reinforcement Learning pada dasarnya mengkloning perilaku permainan dan**

No	Nama Penulis	Judul	Hasil
			memungkinkan agen melakukan dengan cara yang sama ketika menghadapi situasi yang sama. Maka, pemain tidak mengambil *reward* untuk memutusakan tindakan mana yang harus dipilih sendiri. Dibutuhakan pengalaman (sejarah posisi pemain) bagi pemain untuk mengetahui bagaimana melakukan aksi di posisi yang sama. **Hasil yang diperoleh dari beberapa sesi pelatihan bahwa Imitation Learning dibawah pengawasan Reinforcement Learning menjadi solusi yang optimal. Terbukti adalanya pengurangan waktu yang signifikan untuk memproses model dimana kerugian pada grafik Tensorflow terus menurun.**

No	Nama Penulis	Judul	Hasil
7.	**Icaro Goulart, Aline Paes, dan Esteban Clua (2019)**	***Learning how to play Bomberman with Deep Reinforcement and Imitation Learning***	**Penelitian ini Membuat agen buatan yang belajar cara bermain yaitu dengan merancang agen yang mempelajari cara memainkan permainan populer Bomberman dengan mengandalkan representasi status dan algoritma berbasis RL tanpa melihat level piksel. Penelitian ini merancang lima representasi status berbasis vektor dan mengimplementasikan Bomberman di atas mesin *game Unity* melalui *ML-agents toolkit.*** **Penelitian ini meningkatkan algoritme agen ML dengan mengembangkan Imitation Learning (IL) yang meningkatkan modelnya dengan metode Proximal Policy Optimization (PPO). Kami membandingkan pendekatan ini**

No	Nama Penulis	Judul	Hasil
			dengan pelajar PPO saja yang menggunakan Multi-Layer Perceptron atau jaringan Long Short Term-Memory (LSTM). Representasi keadaan hibrida dan IL yang diikuti oleh algoritme pembelajaran PPO mencapai hasil kuantitatif terbaik secara keseluruhan, dan juga agen dapat mempelajari perilaku Bomberman yang benar. **Penelitian ini menyelidiki lima representasi keadaan dan empat algoritma pembelajaran untuk membangun agen Bomberman yang mempelajari cara bermain. Mengenai representasi state, hasil menunjukkan bahwa representasi hibrida yang kami usulkan mencapai hasil terbaik dalam waktu pengujian, bereksperimen dengan turnamen yang**

No	Nama Penulis	Judul	Hasil
			dilakukan setelah fase pembelajaran. Mengenai algoritma pembelajaran, penelitian ini bereksperimen dengan algoritme PPO berbasis aktor-kritikus menggunakan MLP atau LSTM, pembelajaran imitasi BC, dan pendekatan baru yang dikembangkan dimulai dengan BC dan melanjutkan pelatihan dengan PPO dari fungsi yang dipelajari dengan BC. Hasil menunjukkan bahwa dengan menggabungkan LSTM dalam algoritma PPO menghasilkan agen yang lebih cerdas dan pelatihan sebelumnya dengan algoritma BC dapat mempengaruhi pelatihan selanjutnya dengan PPO.

No	Nama Penulis	Judul	Hasil
8.	**Teemu Ropilo (2019)**	***Teaching A Machine Learning Agent To Survive In A 2d Top-Down Environment***	**Penelitian ini bertujuan untuk melatih agen pembelajaran mesin otonom yang dapat bertahan secara top-down pada lingkungan dua dimensi dengan melakukan tugas yang diperlukan untuk mencapai tujuan yang telah ditetapkan. Metode pembelajaran yang berbeda diperiksa dan dibandingkan untuk menemukan metode terbaik untuk melatih agen.** **Proses sebelum lingkungan pelatihan dibuat adalah dengan mengumpulkan referensi terkait pembelajaran mesin terutama di bidang pengembangan *game*, membuat rancangan awal untuk implementasi yang diikuti dengan penjabaran dokumentasi. Pengembangan lingkungan**

No	Nama Penulis	Judul	Hasil
			pelatihan adalah mengkonfigurasi alat pembelajaran mesin dan melakukan tes awal sebagai bukti konsep. **Tahap ini merupakan proses untuk menindaklanjuti perbaikan dan pengecekan setiap elemen. Selama pelatihan agen pembelajaran mesin, statistik dikumpulkan untuk memeriksa efisiensi setiap metode pembelajaran agar menemukan metode pembelajaran terbaik untuk keperluan penelitian.** **Hasil penelitian menunjukan bahwa alat dan metode pembelajaran mesin yang digunakan dalam penelitian ini terbukti menjadi sarana yang dapat diterapkan untuk mencapai tujuan. Pembelajaran peniruan (Imitation**

No	Nama Penulis	Judul	Hasil
			Learning) yang dibantu pembelajaran penguatan (Reinforcement Learning) adalah pilihan terbaik dari metode yang diteliti, meskipun metode lain juga menunjukkan harapan. Langkah selanjutnya untuk mendapatkan hasil yang lebih baik, dilakukan setting pengamatan yang diumpankan ke otak dan menguji hyperparameter yang berbeda. Menjalankan sesi latihan yang lebih lama juga bisa menciptakan hasil yang lebih baik. **Sebagai kesimpulan, dapat dikatakan bahwa model yang dihasilkan mampu memeriksa lingkungan dan mencapai tujuan yang ditetapkan ke tingkat yang dapat diterima, bahkan bisa dikatakan sukses. Mengingat tingkat**

No	Nama Penulis	Judul	Hasil
			pengetahuan subjek pada awal penelitian hasilnya sukses dan jumlah server pengetahuan yang terakumulasi sebagai batu loncatan untuk melakukan pengembangan lebih lanjut.
9.	**Olli-Pekka Juhola (2019)**	***Creating Self-Learning AI using Unity Machine Learning***	**Penelitian dilakukan untuk mengembangkan kecerdasan buatan dalam *game* sederhana menggunakan dua metode terpisah dengan mesin *game* Unity dan membandingkan hasilnya satu sama lain.** **Metode implementasi yang digunakan adalah sistem navigasi dan pencarian jalur mesin *game Unity* sendiri, serta alat yang diproduksi oleh *Unity* yang memungkinkan penggunaan pembelajaran mesin dalam proyek *Unity*. Tujuan utamanya adalah untuk membuat prototipe yang berfungsi**

No	Nama Penulis	Judul	Hasil
			menggunakan keduanya metode untuk mencari dan mengumpulkan koin di lingkungan dengan kecerdasan buatan dan untuk dapat menunjukkan, bagaimana kedua metode tersebut diterapkan.Akibatnya, dua *game* fungsional dikembangkan menggunakan metode AI yang berbeda, dan kedua agen AI mencari dan mengumpulkan koin sesuai keinginan. **Metode pembelajaran mesin menggunakan pembelajaran yang diperkuat dalam pelatihan AI, sehingga agen tidak tahu apa-apa tentang permainan selain cara bergerak. Ia kemudian berlatih secara mandiri di lingkungan ratusan kali per menit untuk mengetahui esensi permainan.**

No	Nama Penulis	Judul	Hasil
			Metode navigasi dan pengaturan jalur *Unity* dibuat dengan sistem jaringan navigasi *Unity*, yang memungkinkan kecerdasan buatan untuk mengerti ke mana Anda bisa bergerak dan mengetahui lokasi koin secara otomatis. **Kedua metode tersebut diadu satu sama lain dalam tes sederhana untuk melihat berapa banyak koin yang dikumpulkan kedua metode AI dalam beberapa menit. Dari koin yang dikumpulkan menggunakan setiap metode, AI yang diajarkan mesin menerima sejumlah besar koin lebih banyak dan mampu mengalahkan sistem navigasi dan jalur mesin *Unity* sendiri. Menggunakan pembelajaran mesin untuk**

No	Nama Penulis	Judul	Hasil
			menciptakan kecerdasan buatan di mesin *game Unity* lebih sulit, tetapi hasilnya jauh lebih menarik.
10.	**Louis-Samuel Pilcer, Antoine Hoorelbeke, Antoine d'Andigne (2018)**	***Playing Flappy Bird with Deep Reinforcement Learning***	**Penelitian ini mengadaptasi penguatan mendalam Deep Q-Network agen pembelajaran untuk FlappyBird, sebuah *game* dengan lingkungan yang sangat sedikit umpan balik menit dan masukan gambar berdimensi tinggi. Kami berlatih dua model untuk mempelajari dampak preprocessing observasi pada kecepatan pelatihan (jumlah iterasi sebelum agen kami memulai mempelajari kebijakan yang efisien). Masalah ini dapat dipelajari lebih lanjut dengan memberikan analisis yang lebih rinci mengenai dampaknya rekayasa penghargaan dan faktor diskon. Kami juga bisa menganalisis dampak dari**

No	Nama Penulis	Judul	Hasil
			nilai maksimal epsilon. Hal menarik lainnya Elemen yang perlu dikaji lebih dalam adalah preprocessing dan pengurangan dimensi, karena mungkin membantu pelatihan lebih cepat dan penemuan fitur. Terakhir, kita mungkin mempelajari dampaknya mentransfer pengetahuan yang dipelajari oleh model yang dilatih pada model lain lingkungan serupa pada kecepatan dan kinerja pelatihan.

2.8 Matriks Penelitian

Berikut ini merupakan Matriks penelitian yang digunakan untuk membandingkan penelitian terkait yang pernah dilakukan sebelumnya dengan penelitian yang akan dilakukan. Matriks penelitian ini disusun agar dapat memudahkan dalam memberikan gambaran mengenai tahapan metode dari *framework* penelitian yang akan dilakukan.

Tabel 2. 3 Matriks Penelitian

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
1.	***Imitation Learning with the Unity Machine Learning Agents Toolkit (A feasibility study to create***	**Mario Rubak (2021)**	**x**	**x**			**x**	**x**

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
	artificial players)							
2.	***Character Animation Using Reinforcement Learning and Imitation***	**Tokey Tahmid, Mohammad Abu Lobabah, Muntasir Ahsan, Raisa Zarin,**	**x**					**x**

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
	Learning Algorithms	**Sabah Shahnoor Anis (2021)**						
3.	***Application of machine learning to construct advanced NPC***	**Johan Fröberg & Carl Håkansson (2021)**	**x**					**x**

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
	behaviors in Unity 3D							
4.	***Training intelligent tennis adversaries using self-play with ML agents***	**Bakhtiyar Ospanov (2021)**	**x**	**x**			**x**	

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
5.	***Applying Imitation and Reinforcement Learning to Sparse Reward Environments***	**Haven Brown (2020)**	**x**		**x**			**x**
6.	***Building your kingdom Imitation***	**Amira E.Youssef, Sohaila El**	**x**	**x**	**x**			

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
	Learning for a Custom Gameplay Using Unity ML-agents	**Missiry, Islam Nabil El-gaafary, Jailan S. ElMosalami, Khaled M. Awad, Khaled Yasser (2019)**						

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
7.	***Learning how to play Bomberman with Deep Reinforcement and Imitation Learning***	**Icaro Goulart, Aline Paes, dan Esteban Clua (2019)**	**x**	**x**			**x**	**x**

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
8.	***Teaching A Machine Learning Agent To Survive In A 2D Top-Down Environment***	**Teemu Ropilo (2019)**	**x**			**x**	**x**	

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
9.	***Creating Self-Learning AI using Unity Machine Learning***	**Olli-Pekka Juhola (2019)**	**x**					**x**
10.	***Playing Flappy Bird with Deep***	**Louis-Samuel Pilcer, Antoine**	**x**		**x**			

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
	Reinforcement Learning	**Hoorelbeke, Antoine d'Andigne (2018)**						
11.	**Implementasi *Machine Learning Agents* Dengan Menggunakan**	**Devi Rostina (2021)**	**x**	**x**	**x**	**x**	**x**	**x**

No	Judul	Peneliti	Perbedaan antara penelitian terdekat					
			Diakses lewat PC	***Unity Machine Learning Agents Toolkit***	**Pengurangan waktu *training* dilihat dari *tensorflow***	**Kemampuan adaptasi Agen tinggi**	**Meningkatkan kemampuan *Learning Agent* setelah proses *learning***	**Menggunakan minimal 2 algoritma sebagai pembanding**
	Deep Reinforcement Learning* Pada *Game Flappy Bird							