IMPLEMENTASI Q-LEARNING YANG DIKOMBINASIKAN DENGAN CONVOLUTIONAL NEURAL NETWORK PADA AGENT YANG MEMAINKAN PERMAINAN FLAPPY BIRD

Fajar Abdi Nugraha[1], Ednawati Rainarli[2]

[1,2] Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-114 Bandung
E-mail : nugrahafajar37@gmail.com[1], ednawati.rainarli@email.unikom.ac.id[2]

ABSTRAK

Penelitian. ini .mengimplementasikan Convolutional Neural Network sebagai action-value function approximation untuk memprediksi nilai Q-value pada Q-Learning untuk setiap action dan diterapkan pada agen yang memainkan permainan flappy bird. Penggunaan action-value function approximation bertujuan untuk mengurangi banyaknya jumlah percobaan pada proses eksplorasi dan meningkatkan skor yang didapat agen saat memainkan permainan, karena berdasarkan hasil penelitian sebelumnya yang menggunakan Q-Learning saja dibutuhkan banyak percobaan memainkan permainan pada proses eksplorasi untuk dapat meningkatkan skor yang didapat agen saat memainkan permainan. Tahapan yang terdapat pada sistem terbagi menjadi 2, yaitu eksplorasi dan eksploitasi. Eksplorasi adalah proses pembelajaran agen dalam memainkan permainan, pengetahuan yang telah didapat pada tahap eksplorasi akan digunakan pada tahap eksploitasi. Tahap eksploitasi adalah proses uji coba dari pengetahuan yang telah didapat pada tahap sebelumnya, proses uji coba dilakukan untuk mengetahui sejauh mana kemampuan agen dalam memainkan permainan. Berdasarkan hasil dari penelitian bahwa implementasi.Q-Learning yang dikombinasikan dengan.CNN menunjukan hasil agen dapat menyelesaikan proses eksplorasi dengan rata-rata 2336,6 percobaan dan skor rata-rata yang didapat adalah 575,2 dimana 1 dari 5 percobaan berhasil menyentuh skor 1000.

Kata kunci : *Convolutional Neural Network* (CNN), *action-value function approximation*, *Q-Learning*, *agen*, *Flappy bird*

1. PENDAHULUAN

Permainan flappy bird termasuk ke dalam kategori gulir-samping, dimainkan dengan cara mengendalikan seekor burung naik atau turun. Permainan ini memiliki tujuan untuk melewati celah antar pipa yang memiliki ketinggian beraneka ragam. Jika pemain berhasil melewati celah tersebut maka skor bertambah, jika pemain menabrak pipa atau dasar tanah maka pemain dinyatakan kalah.

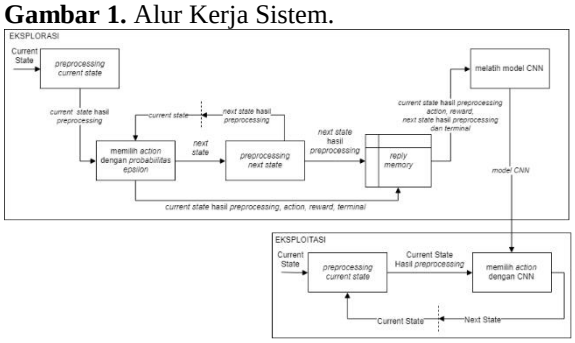
Adapun penelitian penyelesaian permainan flappy bird sebelumnya oleh Moritz [2] menggunakan Q-learning didapatkan hasil, selama 1000kali percobaan skor tertinggi yang didapat berada diangka 169 [1] lebih kecil dibandingkan kemampuan manusia ahli yaitu Ryu Dragon mencapai skor328 [3]. Selain itu, dijelaskan bahwa kemampuan agen dalam mendapatkan skor bergantung pada banyaknya jumlah percobaan pada proses eksplorasi, semakin banyak jumlah percobaan maka kemampuan akan semakin membaik [2]. Hal tersebut terjadi karena flappy bird memiliki komponen bersifat dinamik, yaitu posisi burung, pipa dan celah antar pipa yang beraneka ragam, berakibat state-space menjadi besar dan proses eksplorasi membutuhkan banyak percobaan karena action-value function pada Q-Learning tidak memiliki kemampuan untuk prediksi Q-value suatu action pada state yang baru ditemui [2]. Masalah tersebut dapat diatasi dengan menambah suatu metode untuk mengeneralisasi state [2] sehingga agen bisa memprediksi Q-value suatu action pada state yang belum pernah dihadapi sebelumnya.

Salah satu improvisasinya adalah menambahkan action-value function approximation pada algoritma Q-Lerning [4] untuk memprediksi nilai Q-value setiap action pada state tertentu. Penggunaan metode tersebut dapat mengeneralisasi state, sehingga agen bisa memperkecil jumlah percobaan proses eksplorasi dengan tetap mendapatkan hasil yang baik [4]. Penelitian sebelumnya mengenai penggunaan Convolutional-Neural-Network (CNN) sebagai action-value function approximation pada Q-Learning dilakukan oleh Mnih dan kawan-kawan [4] sebagai pionir pengkombinasian ini untuk kasus permainan Atari-2600. Didapatkan hasil, 3 permainan berhasil melampai kemampuan manusia ahli dari total 7 permainan [5]. Penelitian lainnya oleh Ajay Rao dan kawan-kawan [6] yang menggunakan kombinasi Q-Learning dan CNN pada 8 permainan dari OpenAI-Gym. Penelitian tersebut menjelaskan bahwa penggunaan metode tersebut mampu mengeneralisasi state-space dinamik dan bisa diterapkan pada kasus sistem otonom [5].

Oleh karena itu, pada penelitian ini akan mengimplementasikan metode Q-Learning dikombinasikan dengan CNN untuk kasus uji permainan flappy bird.

2. ISI PENELITIAN

Pada alur kerja dari sistem yang dibangun terdapat dua tahap utama, yaitu tahap eksplorasi dan eksploitasi. Satu kali iteras pada sistem sama dengan satu frame pada permainan. Adapun gambaran alur kerja dari sistem yang dibangun terdapat pada **Gambar 1.** Alur Kerja Sistem.



Gambar 1. Alur Kerja Sistem

Tahap pertama adalah eksplorasi, yaitu tahap agen mengumpulkan pengetahuan memainkan permainan, tahap ini dimulai dari agen mendapat current state dari permainan berupa citra keadaan permaian, lalu dilakukan preprocessing terhadap current state. Preprocessing terdiri dari pemotongan citra, konversi ke grayscale, resize dan thresholding. Setelah preprocessing, agen akan menentukan action dengan cara probabilitas epsilon, setelah action didapat dan dieksekusi di permainan agen akan mendapat reward, next state dan terminal sebagai umpan balik. Lalu, akan dilakukan preprocessing terhadap next state. Setelah itu, masukan current state hasil preprocessing, action, reward, next state dan terminal (disebut experience) ke dalam reply memory dan latih model CNN dengan data latih (experience) yang diambil secara acak dari reply memory sebanyak 32(ukuran batch). Terakhir, lakukan perubahan next state hasil preprocessing menjadi current state dan proses diulangi dari tahap pemilihan action dengan probabilitas epsilon. Tahap ini memiliki kondisi henti yaitu ketika agen menyentuh skor di angka 20 atau saat agen sudah memainkan permainan selama 1000000 frame.

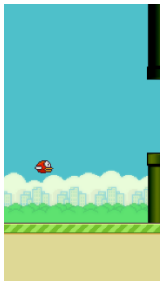
Tahap kedua adalah eksploitasi, yaitu tahap mengujikan model CNN yang telah dilatih pada tahap eksploitasi. Tahap ini diawal dengan agen mendapat current state dari permainan, lalu current state tersebut dilakukan preprocessing yang tahapannya sama dengan yang sebelumnya. Selanjutnya agen akan mengambil action menggunakan model CNN dengan input current state yang telah dipreprocessing. Output dari model CNN adalah nilai Q-value dari setiap action yang bisa dilakukan agen. Setelah Q-value dari setiap action didapatkan, agen akan memilih action dengan Q-value terbesar. Pada tahap eskploitasi skor maksimal yang digunakan adalah skor 1000.

2.1. Data Masukan

Ada beberapa data masukan yang digunakan pada metode ini, antara lain:

2.1.1. State

State adalah citra dari keadaan permainan flappy bird. Ada dua jenis state, yaitu current state yang merupakan keadaan saat ini atau keadaan sebelum action dieksekusi dan next state adalah keadaan setelah action dieksekusi. Contoh *state* dapat dilihat pada **Gambar 2.** State Permainan.



Gambar 2. State Permainan

2.1.2. Action

Terdapat 2 action tersedia yang dapat dilakukan agen untuk environment permainan flappy bird, setiap action tersedia akan diberi nilai sebagai tanda. Berikut adalah action yang dapat dilakukan:

1. Action bernilai 0 untuk turun terhadap burung
2. Action bernilai 1 untuk naik terhadap burung

2.1.3. Reward

Reward adalah umpan balik dari permainan terhadap action yang dilakukan agen pada state tertentu berupa nilai. Mengenai aturan reward dipakai ,antara lain:

1. +1 disaat burung melewati celah antar pipa.
2. 0,1 disaat burung tetap hidup.
3. -1 disaat burung menyentuh pipa atau tanah.

2.1.4. Terminal

Terminal adalah status pemain di permainan setelah action dieksekusi, dengan tipe data boolean. Akan bernila True jika setelah action dieksekusi permainan berakhir, dalam flappy bird permainan berakhir jika burung menyentuh pipa atau dasar tanah. Bernilai false jika burung masih bertahan hidup setelah action dieksekusi.

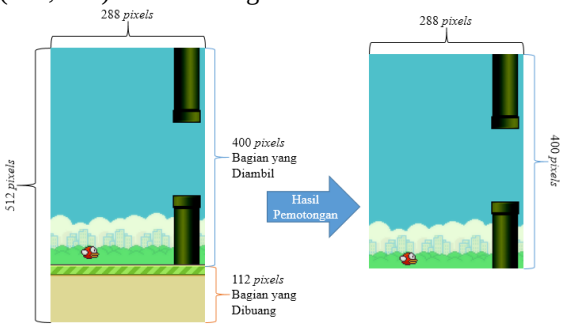
2.2. Preprocessing State

Preprocessing adalah proses mengolah data masukan(state) sebelum digunakan pada proses utama. Masukan data asli berbentuk citra dari keadaan permainan. Citra tersebut akan melalui proses preprocessing meliputi pemotongan, konversi citra RGB ke grayscale, resize dan thresholding.

2.2.1. Pemotongan

Pemotongan adalah proses menyisihkan sebagian bagian dari citra dengan koordinat tertentu. Pada proses ini citra asal yang memiliki ukuran 288x 512 akan dilakukan pemotongan pada sumbu Y dari koordinat 401 sampai 512 sedangkan sumbu X tidak

akan dilakukan pemotongan. Sehingga didapatkan koordinat pojok kiri atas (0, 0) dan kanan bawah (288, 400). Maka citra yang terdapat pada koordinat (0, 0) sampai (288, 400) akan dipertahankan dan citra dengan koordinat (0, 401) sampai koordinat (288, 512) akan dibuang.



Gambar 3. Ilustrasi Pemotongan

2.2.2. Grayscale

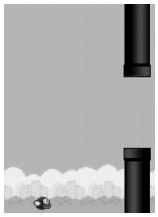
Citra yang menjadi masukan awalnya memiliki 3 chanel warna yaitu red, blue dan green. Pada proses konversi citra RGB ke grayscale, chanel pada citra akan berubah menjadi 1 saja dan diharapkan dapat memperingan beban komputasi [6]. Perhitungan konversi citra RGB ke grayscale dapat menggunakan persamaan di bawah ini.

$$I = 0.299*R + 0.587*G + 0.114*B \quad (1)$$

Keterangan:

I = nilai grayscale dari hasil perhitungan
R = nilai untuk chanel red pada pixel
G = nilai untuk chanel green pada pixel
B = nilai untuk chanel blue pada pixel

Berikut adalah citra hasil konversi ke grayscale



Gambar 4. Citra Hasil Grayscale

2.2.3. Resize

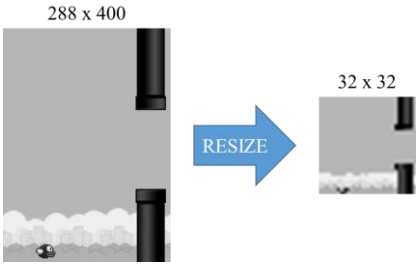
Resize akan merubah ukuran citra sesuai dengan yang dikehendaki, dapat memperkecil atau memperbesar citra, pada penelitian ini citra akan di resize menjadi lebih kecil dari ukuran awal 288x512 pixels, menjadi 32x32 pixels. Pada.proses penurunan ukuran akan diambil nilai tengah dari suatu kernel untuk mewakili kernel tersebut [9]. Penentuan ukuran kernel diambil dari hasil bagi ukuran citra awal dengan citra hasil. Perhitungan penurunan citra menggunakan metode interpolasi bilinier yang merupakan lanjutan dari interpolasi linear untuk sumbu atau variabel ganda [6]. Cara kerjanya adalah menggunakan interpolasi linier terlebih dahulu pada satu sumbu kemudian pada sumbu lainnya. Adapun persamaan interpolasi sebagai berikut.

$$f(x,y) = \frac{y2-y}{y2-y1}.f(x,y1) + = \frac{y-y1}{y2-y1}.f(x,y2) \quad (2)$$

Keterangan:

f(x, y) = nilai pixel hasil interpolasi
x = koordinat sumbu X yang ingin diketahui
y = koordinat sumbu Y yang ingin diketahui
y1 = koordinat sumbu Y yang telah diketahui nilainya dan beradar setelah koordinat interpolasi
y2 = koordinat sumbu Y yang telah diketahui nilainya dan beradar sebelum koordinat interpolasi

Ilustrasi hasil resize citra dapat dilihat dibawah ini.



Gambar 5. Ilustrasi Resize

2.2.4. Thresholding

Thresholding adalah metode yang mengubah nilai dari suatu matrik menjadi hanya dua nilai saja. Nilai pembatas disebut ambang batas, penelitian ini akan menggunakan metode thresh binary inverted dengan ambang batas 127, nilai maksimal 255 dan minimal 0, berikut adalah persamaanya.

$$(x,y) = \begin{cases} 0, & img(x,y) > 127 \\ 255, & img(x,y) \leq 127 \end{cases} \quad (3)$$

Ilustrasi perubahan citra hasil thresholding dapat dilihat dibawah ini.

Gambar 6. Ilustrasi Thresholding

2.3. Pemilihan Action dengan Probabilitas Epsilon

Proses pemilihan action dengan probabilitas epsilon terdiri dari beberapa tahapan, berikut adalah gambaran tahapannya.

Gambar 7. Proses Memilih Action dengan Probabilitas Epsilon

Proses dimulai dengan insialisasi nilai mula epsilon yaitu 1.0, akhir epsilon 0.0001 dan pengurangan epsilon sebesar 0.0000198, yang akan terus mengurangi epsilon di setiap framenya. Selanjutnya ambil nilai random antara 0 sampai.1 dan komparasikan dengan nilai epsilon. Jika nilai random lebih kecil dari epsilon, maka agen akan menentukan action acak antara 0(turn) atau 1(naik).

Jika nilai random lebih besar dari epsilon, maka agen akan menentukan action dengan nilai Q-value terbesar atau maksimal yang didapat dari output feedforward CNN dengan input-an current state.

2.4. Metode Q-Learning Dikombinasikan CNN

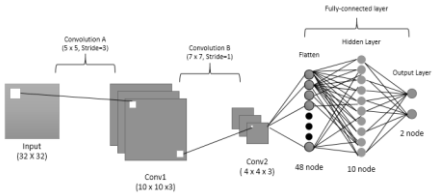
Pada metode pengkombinasian, CNN digunakan sebagai action-value function approximation pada Q-Learning untuk memprediksi nilai Q-value setiap action pada current state dan agen akan menentukan action yang akan diambil dengan membandingkan Q-value tiap action, action dengan Q-value terbesar atau maksiamal yang akan dipilih. Pada proses pelatihan CNN, target yang akan digunakan untuk menghitung nilai loss dan perhitungan gradien adalah persamaan value-iteration-update yang terdapat pada metode Q-learning, berikut targetnya.

$$Set\ Y' = \begin{cases} r, & terminal = True \\ r + \gamma\ max_a Q(S_{t+1}, a'; \theta), & terminal = False \end{cases} \quad (4)$$

Keterangan:

Y'	= target dari pelatihan CNN
r	= reward yang didapat agen
γ	= *Discount rate*, $0 \leq \gamma \leq 1$,
$max_a Q(S_{t+1}, a'; \theta)$	= nilai hasil prediksi atau proses feedforward CNN pada next state dengan nilai output yang maksimal
Terminal	= Status permainan pada next state

Adapun arsitektur dari CNN yang digunakan pada penelitian ini, sebagai berikut.



Gambar 8. Arsitektur CNN

2.4.1. Feedforward

Pada tahap feedforward akan dilakukan proses CNN dari awal masukan(state hasil preprocessing) melewati convolutional layer A, convolutional layer B dan fully-connected layer hingga dihasilkan 2 buah output yang berisi Q-Value untuk masing-masing action yang bisa agen lakukan.

a. Convolution Layer A

Convolutional layer terdiri dari kumpulan neuron membentuk filter. Operasi yang terjadi pada layer adalah dot product antara filter dengan matriks citra inputan, filter akan terus digeser sebesar stride hingga seluruh permukaan matriks. Hasil operasi tersebut disebut feature feature map. Pada Convolution Layer A terdapat 3 buah filter berukuran 5x5 dengan stride 3, berikut adalah persamaan dari operasi yang terjadi pada filter A.

$$\boldsymbol{FA1_{i,j}} = \sum_{m=0}^{4} \sum_{n=0}^{4} \boldsymbol{A1_{m,n}}\ \boldsymbol{C_{i*3+m,j*3+n}} + \boldsymbol{bA_0} \quad (5)$$

Keterangan:

FA1	= matriks feature map hasil kalkulasi
A1	= matriks filter A1
C	= matriks input citra
bA	= niali bias pada layer A
i	= nilai penunjuk baris feature map
j	= nilai penunjuk kolom feature map
m	= nilai penunjuk baris pada filter
n	= nilai penunjuk kolom pada filter

Setelah mendapat matriks feature map FA1, lalu aktifkan fungsi aktivasi ReLU pada setiap neuron.

$$\boldsymbol{RA1(x, y) = max(0, RA1(x, y)} \quad (6)$$

Keterangan:

RA1	= feature map hasil pengaktifan ReLU

Ketika diaktifkan Fungsi tersebut akan mengubah setiap nilai negative pada matriks menjadi 0.

b. Convolution Layer B

Pada convolutional layer B akan dilakukan proses yang sama seperti pada convolutinal layer A hanya saja inputannya rangkap tiga, yaitu feature map RA1, RA2 dan RA3. Pada convolutional layer B terdapat 3 filter berukuran 7x7 dengan stride 1. Berikut persamaanya.

$$\boldsymbol{FB1_{i,j}} = \sum_{C=1}^{3} \sum_{m=0}^{6} \sum_{n=0}^{6} \boldsymbol{B1_{m,n}} \cdot \boldsymbol{RAc_{i*1+m,j*1+n}} + \boldsymbol{bB_0} \quad (7)$$

Keterangan:

FB1	= feature map hasil operasi Filter B1
B1	= nilai matrks filter B1
RAc	= input filter RA rankap ke-c
bB	= nilai bias pada layer B
i	= nilai penunjuk baris feature map
j	= nilai penunjuk kolom feature map
m	= nilai penunjuk barispada filter
n	= nilai penunjuk kolompada filter

Setelah mendapat feature map FA1, selanjutnya aktifkan fungsi yang sama dengan layer A, yaitu bernama ReLU dan menghasilkan feature map FB. Pada layer ini menhasilkan 3 feature map berukuran 4x4, yang selanjutnya akan dimasukkan ke fully-connected layer.

c. Fully Connected Layer

Fully-Connected-Layer memiliki input ,hidden dan output layer selayaknya percepton-layer-ganda. Berfungsi untuk melakukan regresi state masukan menjadi Q-value tiap action, berikut layer-layernya.

- Input layer (flatten)

  Pada input-layer akan terjadi proses meluruskan inputan berupa feature map 4x4 berjumlah 3 sehingga jumlah node pada input layer adalah 48 buah node.

- Hidden layer

  Pada hidden-layer dari arsitektur yang dipakai memiliki 10 buah node, 48 node pada input akan terhubung seluruhnya dengan 10 node hidden-layer. Sehingga total parameter yang terdapat adalah 480 buah, berikut adalah persamaan perhitungannya.

$$z\_in_i = \sum_{j=0}^{n=47} X_j * V_{j,i} + bV_i \quad (8)$$

Keterangan:

z_in_i = nilai input node hidden layer
X_j = nilai untuk node X diposisi ke j
$V_{j,i}$ = nilai bobot V di posisi ke j dan ke i
bV = nilai bias pada hidden layer V

Lalu kalkulasikan nilai z_in, yaitu hasil proses dari node hidden-layer. Setelah itu kalkulasi nilai keluaran Z dengan mengaktifkan fungsi aktivasi ReLU pada tiap-tiap nilai pada matriks z_in.

- Output layer

Output-layer memiliki2 buah node output (sejumlah action yang bisa dilakukan agen), setiap node terhubung dengan output-layer Z. Terjadi operasi dengan menggunakan persamaan berikut.

$$y\_in_i = \sum_{j=0}^{n=9} Z_j * W_{j,i} + bW_i \quad (9)$$

Keterangan:

y_in_i = masukan untuk node output-layer Y
Z_j = nilai untuk node Z di posisi ke-j
$W_{j,i}$ = nilai parameter Wdi posisi ke-j, ke-i
bW_i = nilai bias di node ke-i
n = jumlah seluruh node hidden-layer Z

Setelah mengetahui matriks y_in selanjutnya aktifkan fungsi aktivasi linear untuk mengeluarkan nilaiY. Berikut persamaan fungsi aktivasi linear.

$$\varphi(x) = x \quad (10)$$

Ketrangan

X = inputan pada fungsi
φ = simbol untuk fungsi aktivasi

Output-layer terdapat 2 output(sejumlah action, 0 dan 1), nilai yang dihasilkan dari feedforward adalah Q-Value dari masing-masing action.

2.4.2. Backpropagation

Pada tahap ini akan dilakukan kalkulasi gradien dari tiap-tiap parameter dan koreksi parameter berdasarkan nilai error. Nilai error yang berasal dari selisih hasil prediksi(feedforward) dengan target yang ditentukan. Proses backpropagation terdiri dari beberapa perhitungn, berikut alur perhitungannya.

a. Perhitungan Loss

Loss function adalah fungsi untuk mengkalkuasi nilai error dari prediksi suatu model terhadap target, pada penelitian ini akan menggunakan Mean Squared Error (MSE). Berikut persamaanya

$$L = \frac{1}{n}\sum_i^n (Y_i - Y'_i)^2 \quad (11)$$

Keterangan:

L =Nilai hasil kalukasi loss dengan MSE
N =Banyaknya data dari output model
Y =Nilai hasil feedforward dari model
Y' =Nilai target pelatihan dari data latih

b. Perhitungan Gradien Output layer

Setelah perhitungan nilai loss, selanjutnya hitung gradien parameter-W terhadap Loss, dengan rumus chain-rule.

$$\frac{\partial L}{\partial W_{j,i}} = \frac{\partial L}{\partial Y_i}\frac{\partial Y_i}{\partial y_{in_i}}\frac{\partial y_{in_i}}{\partial W_{j,i}} \quad (12)$$

Keterangan:

$\frac{\partial L}{\partial W_{j,i}}$ = Gradien W terhadap nilai *loss*
$\frac{\partial L}{\partial Y_i}$ = Turunan parsial fungsi loss MSE
$\frac{\partial Y_i}{\partial y_{in_i}}$ = Turunan fungsi aktivasiLinear, yaitu 1
$\frac{\partial y_{in_i}}{\partial W_{j,i}}$ = Gradien W terhadap y_in, yaitu nilai Z

c. Perhitungan Gradien Hidden Layer

Selanjutnya hitung gradien parameter W terhadap nilai Loss, masih dengan rumus chain-rule sebagai berikut.

$$\frac{\partial L}{\partial V_{k,j}} = \sum_i^m \frac{\partial L}{\partial Y_i}\frac{\partial Y_i}{\partial y_{in_i}}\frac{\partial y_{in_i}}{\partial Z_j}\frac{\partial Z_j}{\partial z\_in_j}\frac{\partial z\_in_j}{\partial V_{k,j}} \quad (13)$$

Keterangan:

$\frac{\partial L}{\partial V_{k,j}}$ = Gradien parameter V terhadap *loss* L
$\frac{\partial y_{in_i}}{\partial Z_{j,i}}$ = Gradien Z terhadap yin_i, yaitu $W_{j,i}$
$\frac{\partial Z_j}{\partial z\_in_j}$ = turunan fungsi.aktivasi ReLU terhadap z_in, yaitu $\begin{cases} 1 & z\_in_j \geq 0 \\ 0 & z\_in_j < 0 \end{cases}$
$\frac{\partial z\_in_j}{\partial V_{k,j}}$ = Gradien V terhadap z_in, yaitu X_k

d. Perhitungan Gradien Convolution Layer B

Selanjutnya hitung gradien parameter filter B terhadap loss, menggunakan chain rule berikut.

$$\frac{\partial L}{\partial Bi} = \frac{\partial L}{\partial Xi}\frac{\partial Xi}{\partial FBi}\frac{\partial FBi}{\partial Bi} \quad (14)$$

Sebelum menghitung gradien filter B, hitung dahulu $\frac{\partial L}{\partial X_k}$, menggunakan rumus chain rule, sebagai berikut.

$$\frac{\partial L}{\partial X_k} = \sum_j^n \sum_i^m \frac{\partial L}{\partial Y_i}\frac{\partial Y_i}{\partial y_{in_i}}\frac{\partial y_{in_i}}{\partial Z_j}\frac{\partial Z_j}{\partial z\_in_j}\frac{\partial z\_in_j}{\partial X_k} \quad (15)$$

Keterangan:

$\frac{\partial L}{\partial X_k}$ = Gradien matriks X di posisi k terhadap niali loss L
$\frac{\partial z\_in_j}{\partial X_k}$ = Gradien X terhadap z_in_j, yaitu $V_{k,j}$

setelah $\frac{\partial L}{\partial X_k}$, hitung nilai $\frac{\partial L}{\partial FBi}$ dengan rumus berikut.

$$\frac{\partial L}{\partial FBi} = \frac{\partial L}{\partial Xi}\frac{\partial Xi}{\partial FBi} \quad (16)$$

Keterangan:

$\frac{\partial L}{\partial FBi}$ = Gradien feature.map FB ke i.terhadap nilai loss L
$\frac{\partial L}{\partial X_k}$ = Gradien matrisk X di posisi k terhadap loss L
$\frac{\partial Xi}{\partial FBi}$ = turunan fungsi ReLU terhadap FB di posisi ke i, yaitu $\begin{cases} 1 & FB1_{0,0} \geq 0 \\ 0 & FB1_{0,0} < 0 \end{cases}$

Setelah $\frac{\partial L}{\partial FB}$ diketahui, lakukan konvolusi antara $\frac{\partial L}{\partial FB1}$ dengan *featuure map* RA1, RA2 dan RA3 untuk mendapatkan gradien $\frac{\partial L}{\partial B}$, berikut persamaanya.

$$\frac{\partial L}{\partial B1_{i,j}} = \sum_{C=1}^3 \sum_{m=0}^3 \sum_{n=0}^3 \frac{\partial L}{\partial FB1_{m,n}} . \ RAc_{i*1+m,j*1+n} \quad (16)$$

Keterangan:

$\frac{\partial L}{\partial B1_{i,j}}$ = Gradien Filter B1 terhadap nilai loss
$\frac{\partial L}{\partial FBi}$ = Gradien feature map FB terhadap loss
RAc = Featuremap RA rangkap ke-c

e. Perhitungan Gradien Convolution Layer A

Terakhir perhitungan gradien terhadap parameter filter A menggunakan rumus *chain* rule.

$$\frac{\partial L}{\partial Ai} = \sum_{j=1}^{n} \frac{\partial L}{\partial RAj} \frac{\partial RAi}{\partial FAi} \frac{\partial FAi}{\partial Ai} \quad (17)$$

Sebelum menghitung gradien filter A, hitung dahulu $\frac{\partial L}{\partial RAj}$, menggunakan rumus chain rule dengan melakukan *cross-correlation* antara $\frac{\partial L}{\partial FBi}$ dengan Filter Bi. Berikut persamaanya.

$$\frac{\partial L}{\partial RA1_{i,j}} = \sum_{m=0}^{-3} \sum_{n=0}^{-3} \boldsymbol{B1}_{i+m,j+n} \frac{\partial L}{\partial FB1_{-m,-n}} \quad (18)$$

Keterangan:

$\frac{\partial L}{\partial RA1_{i,j}}$ = Gradien RA terhadap loss L di i ke j

$\boldsymbol{B1}_{i,j}$ = nilai Filter B1 di posisi ke i ke j

$\frac{\partial L}{\partial FBi}$ = Gradien FB terhadap nilai loss *L*

Setelah mendapat matriks $\frac{\partial L}{\partial RAi}$. Selanjutnya melakukan perhitungan nilai $\frac{\partial L}{\partial FAi}$.

$$\frac{\partial L}{\partial FAi} = \sum_{j=1}^{3} \frac{\partial L}{\partial RAj} \frac{\partial RAi}{\partial FAi} \quad (19)$$

Keterangan:

$\frac{\partial L}{\partial FAi}$ = Gradien FA ke i terhadap nilai loss L

$\frac{\partial L}{\partial RAj}$ = Gradien RA ke j terhadap nilai loss L

$\frac{\partial RAi}{\partial FAi}$ = turunan fungsi ReLU terhadap FA ke-i, yaitu $\begin{cases} 1 & FA1_{0,0} \geq 0 \\ 0 & FA1_{0,0} < 0 \end{cases}$

Setelah nilai diketahui $\frac{\partial L}{\partial FAi}$, lakukan proses konvolusi antara $\frac{\partial L}{\partial FAi}$ dengan citra input C untuk menghasilkan gradien $\frac{\partial L}{\partial Ai}$. Berikut persamaanya.

$$\frac{\partial L}{\partial A1_{i,j}} = \sum_{m=0}^{27} \sum_{n=0}^{27} \frac{\partial L}{\partial FA1_{m,n}} \cdot \boldsymbol{C}_{i*1+m,j*1+n} \quad (20)$$

Keterangan:

$\frac{\partial L}{\partial A1_{i,j}}$ = Gradien Filter A terhadap nilai loss L

$\frac{\partial L}{\partial FA1}$ = Gradien FA1 terhadap nilai loss L

$\boldsymbol{C}$ = Citra inputan

f. Update Bobot

Proses pembaharuan bobot akan menggunakan metodeAdam yang berdasar pada nilai gradien yang telah dikalkulasi. Adam sendiri adalah metode optimasi-bobot dengan basis gradien orde kesatu dari *stochastic-objective-functions*, berdasarkan estimasi adaptif momen orde rendah, memiliki persamaan berkut.

$$\boldsymbol{m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t} \quad (21)$$

$$\boldsymbol{v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * g_t^2} \quad (22)$$

$$\boldsymbol{\hat{m}_t = m_t / (1 - \beta_1^t)} \quad (23)$$

$$\boldsymbol{\hat{v}_t = v_t / (1 - \beta_2^t)} \quad (24)$$

$$\boldsymbol{\theta_t = \theta_{t-1} - \alpha * m'_t / (\sqrt{v'_t} + \epsilon)} \quad (25)$$

Keterangan:

β_1 = pengurangan momentum ke-1, beta1

β_2 = pengurangan momentum ke-2, beta2

g_t = nilai gradien suatu parameter ke-t

m_t = moment vektor kesatu di-t, dimana $m_0 = 0$

v_t = moment vektor kedua di-t, dimana $v_0 = 0$

$\hat{m}_t$ = estimasi vektor bias-corrected dari Moment kesatu di-t.

$\hat{v}_t$ = estimasi bias-corrected vector Moment kedua di-t

α = lajupembelajaran, kode nama alpha

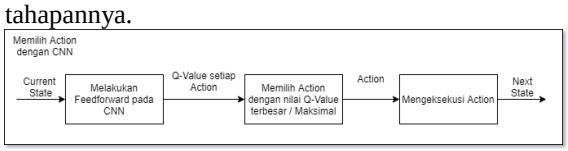
θ_{t-1} = nilai bobot lama sebelum di-t

θ_t = nilai bobot bobot baru hasil kalkulasi

ϵ = epsilon pencegah pembagi 0,umunya 10^{-8}

2.5. Pemilihan Action dengan CNN

Proses pemilihan action dengan CNN terdapat beberapa tahap, berikut adalah gambaran tahapannya.



Gambar 9. Proses Memilih Action dengan CNN

Proses dawali dengan feedforward pada model CNN dengan input-an current state, output dari feedforward adalah nilai Q-value untuk setiap action (0 dan 1). Selanjutnya agen akan memutuskan action yang akan diambil dengan membandingkan nilai Q-value dari seluruh action, action dengan nilai Q-value terbesar akan menjadi action yang dipilih dan selanjutnya dieksekusi di permainan flappy bird. Berikut adalah ilustrasi proses pemilhan action dengan CNN.

Gambar 10. Ilustrasi Pemilihan Action dengan CNN

2.6. Pengujian Peformansi

Pengujian performansi dilakukan untuk mengukur peform dari algoritma yang diterapkan dan melihat pengaruh dari setiap perubahan parameter algoritma untuk mencari kombinasi parameter yang optimal sehingga dapat diketahui rata-rata jumlah percobaan terkecil yang dibutuhkan pada proses eksplorasi dan skor terbesar yang dapat diraih agen. Sebagai catatan pada proses pengujian posisi pipa random dan tidak di atur sama antara satu percobaan dengan percobaan lainnya. Berikut adalah skenario pengujian peformansi.

Tabel 1. Skenario Pengujian

Parameter	Nilai
Learning Rate	0.1, 0.01, 0.001
Discount Rate	0.1, 0.5, 0.99
Ukuran Batch	8, 16, 32
Kondisi Henti	10, 15, 20

2.6.1. Pengujian Learning Rate

Pada pengujian ini menggunakan tiga nilai, yaitu 0.001, 0.01 dan 0.1, untuk parameter lainnya adalah discount rate 0.99, ukuran batch 32 dan kondisi henti di skor 20. Percobaan yang gagal memenuhi kondisi henti ditandai bintang(*) pada isi sel table dengan kolom jumlah percobaan proses eksplorasi.

Tabel 2. Hasil Pengujian Learning Rate

No	0,001		0,01		0,1	
	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat
1	323 percobaan	892	*7420 percobaan	11	*6279 percobaan	4
2	222 percobaan	1000	*7558 percobaan	22	*7005 percobaan	2
3	*4797 percobaan	5	*4336 percobaan	14	348 percobaan	351
4	*5417 percobaan	5	576 percobaan	724	*4843 percobaan	13
5	924 percobaan	974	*4851 percobaan	15	*6418 percobaan	8
Rata-rata	2336,6 percobaan	575,2	4948,2 percobaan	157,2	4978,6 percobaan	75,6

Berdasarkan tabel pengujian, nilai 0,001 memiliki hasil terbaik, dimana eksplorasi selesai dengan rata-rata 2336,6 percobaan dan rata-rata skor 575,2. Selain itu, percobaan yang gagal memenuhi.kondisi henti selalu mendapat skor yang buruk. Hal tersebut mennjukan penggunaan learning-rate yang sangat kecil memperbesar jaminan proses pembelajaran menemukan hasil yang optimum. Sedangkan learning-rate yang terlalu besar akan memperbesar kemungkinan hasil pembelajaran tidak konvergen dan memberikan hasil yang buruk [7].

2.6.2. Pengujian Discount Rate

Pada pengujian ini menggunakan tiga nilai, yaitu 0.1, 0.5 dan 0.9, untuk parameter lainnya menggunakan learning rate 0.1, ukuran batch 32 dan kondisi henti di skor 20. Percobaan yang gagal memenuhi kondisi henti ditandai (*) pada isi sel dengan kolom jumlah percobaan proses eksplorasi.

Tabel 3. Hasil Pengujian Discount Rate

No	0,1		0,5		0,99	
	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat
1	*11203 percobaan	1	*4342 percobaan	4	323 percobaan	892
2	*8422 percobaan	0	*5717 percobaan	12	222 percobaan	1000
3	*9257 percobaan	2	*4372 percobaan	5	*4797 percobaan	5
4	*9510 percobaan	1	*4290 percobaan	5	*5417 percobaan	5
5	*11048 percobaan	1	*4088 percobaan	7	924 percobaan	974
Rata-rata	9888 percobaan	1	4561,8 percobaan	6,6	2336,6 percobaan	575,2

Tabel hasil pengujian menunjukan penggunaan 0,99 memiliki hasil terbaik dimana 3 dari 5 percobaan berhasil memenuhi kondisi henti dengan rata-rata 2336,6 percobaan dan rata-rata skor575,2. Selain itu, asumsi pada pengujian sebelumnya terjadi, yaitu percobaan yang gagal memenuhi kondisi henti cenderung memberikan skor yang buruk. Jika dilihat keseluruhan, penggunaan discount-rate semakin kecil cenderung memberikan hasil yang semakin buruk. Hal ini menunjukkan penggunaan nilai discount-rate yang smendekati 1 akan memberikan hasil optimum dilihat dari skor didapatkan agen [8].

2.6.3. Pengujian Ukuran Batch

Pada pengujian ini menggunakan tiga nilai, yaitu 8, 16 dan 32, untuk parameter lainnya menggunakan learning rate 0.1, discount rate 0.99 dan kondisi henti di skor 20. Percobaan yang gagal memenuhi kondisi henti ditandai bintang(*) pada isi sel table dengan kolom jumlah percobaan proses eksplorasi.

Tabel 4. Hasil Pengujian Ukuran Batch

No	8		16		32	
	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat
1	*4187 percobaan	0	873 percobaan	83	323 percobaan	892
2	*5968 percobaan	6	1053 percobaan	937	222 percobaan	1000
3	*5244 percobaan	7	*6580 percobaan	6	*4797 percobaan	5
4	*7513 percobaan	3	*8478 percobaan	0	*5417 percobaan	5
5	*6378 percobaan	21	*7069 percobaan	13	924 percobaan	974
Rata-rata	5858 percobaan	7,4	4810,6 percobaan	207,8	2336,6 percobaan	575,2

Tabel hasil pengujian menunjukan penggunaan nilai 32 memiliki hasil terbaik, baik dari perspektif rata-rata jumlah percobaan pada proses eksplorasi maupun skor yang didapat. Semakin kecil ukuran batch yang digunakan hasil yang didapat cenderung memburuk. Selain itu, di percobaan ke-1 dengan ukuran batch 16 dimana agen berhasil memenuhi kondisi henti di percoban ke-873 namun skor yang didapat cukup jauh dikomparasikan dengan percobaan lain yang mampu memenuhi kondisi henti. Hasil tersebut menunjukkan ukuran batch berpengaruh terhadap stabilitas proses eksplorasi dan akibatnya mampu menurunkan kemampuan agen, hal tersebut sesuai dengan penelitian J.Lin mengenai kaitan ukuran batch dengan stabilitas proses pembelajaran [9].

2.6.4. Pengujian Kondisi Henti

Pada pengujian ini menggunakan tiga nilai, yaitu 10, 15 dan 20, untuk parameter lainnya menggunakan learning rate 0.1, discount rate 0.99 ukuran batch 32. Percobaan yang gagal memenuhi kondisi henti ditandai bintang(*) pada isi sel table.

Tabel 5. Hasil Pengujian Kondisi Henti

No	10		15		20	
	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat	Jumlah Percobaan Proses Eksplorasi	Skor yang Didapat
1	76 percobaan	11	202 percobaan	1000	323 percobaan	892
2	1014 percobaan	2	701 percobaan	61	222 percobaan	1000
3	129 percobaan	1	165 percobaan	1000	*4797 percobaan	5
4	1212 percobaan	20	*4914 percobaan	5	*5417 percobaan	5
5	373 percobaan	21	*7408 percobaan	8	924 percobaan	974
Rata-rata	560,8 percobaan	11	2678 percobaan	414,8	2336,6 percobaan	575,2

Tabek hasil pengujian menunjukan penggunaan kondisi henti 10 menunjukkan hasil yang baik dengan rata-rata 560,8kali percobaan dan selalu berhasil memenuhi kondisi henti. Namun, kondisi henti 10 justru menunjukkan terburuk dari perspektif skor yang didapat, yaitu rata-rata diangka11. Hal tersebut menunjukkan kondisi henti yang terlalu dini bisa mempercepat proses eksplorasi namun dapat berakibat pada skor yang didapat tidak terlalu baik. Sehingga pemilihan kondisi henti proses eksplorasi akan sangat mempengaruhi terhadap kemampuan agen dalam mendapatkan skor saat memainkan permainan [11].

3. PENUTUP

Pada bagian ini berisi hasil dari penelitian, yaitu kesimpulan dan saran.

3.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan mengenai Implementasi algoritma Q-Learning yang dikombinasikan dengan CNN untuk kasus agen memainkan flappy bird, didapatkan hasil rata-rata jumlah percobaan yang dibutuhkan untuk menyelesaikan proses eksplorasi adalah 2336,6 percobaan dan rata-rata skor yang didapat adalah 575,2. Namun, proses eksplorasi tidak konsisten dimana agen tidak selalu berhasil memenuhi kondisi henti dan banyaknya jumlah percobaan.saat proses eksplorasi tidak selalu berjalan lurus dengan skor yang didapat agen saat proses eksploitasi.

3.2. Saran

Dalam penelitian ini terdapat banyak kekurangan yang terjadi. adapun saran-saran yang dapat diberikan untuk pengembangan pada penelitian selanjutnya agar menjadi lebih baik, antara lain:

1. Menambahkan suatu metode untuk inisialisasi nilai bobot awal di setiap layer.
2. Mencadangkan bobot yang digunakan sebagai checkpoint saat proses eksplorasi tidak berjalan dengan baik.
3. Mencoba membandingkan metode lainnya dengan CNN untuk mengetahui metode terbaik untuk kasus permainan flappy bird.

DAFTAR PUSTAKA

[1] M. Ebeling-Rump, M. Kao dan Z. Hervieux-Moore, "Applying Q-Learning to Flappy Bird," *Department Of Mathematics And Statistics, Queen's University.*

[2] T. M. Buffalo, "Flappy Bird World Record," Tech Marketing Buffalo, 27 February 2014. [Online]. Available: https://techmarketingbuffalo.com/flappy-bird-world-record/.

[3] R. S. Sutton dan A. G. Barto, Reinforcement Learning An Introduction 2nd Edition, Cambridge: The MIT press, 2018.

[4] V. Mnih, K. Kavukcuoglu dan et al, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.

[5] P. A. Rao, B. N. Kumar dan et al, "Distributed Deep Reinforcement Learning using TensorFlow," dalam *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, IEEE, 2017, pp. 171-174.

[6] G. Bradski dan A. Kaehler, Learning OpenCV, Ebastopol: O'Reilly, 2008.

[7] S. Haykin, Neural Networks and Learning Machines, New York: Prentice Hall, 2009.

[8] C. J. Watkins dan P. Dayan, "Q-Learning," *Machine Learning,* vol. 8, no. 3-4, pp. 279-292, 1992.

[9] R. Liu dan J. Zou, "The Effects of Memory Replay in Reinforcement Learning," *arXiv preprint arXiv:1710.06574v1,* 2017.

[10] Adriansyah dan E. Rainarli, "Implementasi Q-Learning dan Backpropagation pada Agen yang Memainkan permainan Flappy Bird," *JNTETI,* vol. 6, no. 1, pp. 1-7, 2017.