Implementasi Cosine Similarity dalam Sistem Matchmaking pada Game Online

Darrel Adinarya Sunanda - 13523061[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*13523061@std.stei.itb.ac.id*, [2]*darrelyanuar@gmail.com*

Abstrak*—Sistem matchmaking merupakan komponen penting dalam game online untuk menciptakan pengalaman bermain yang adil dan menyenangkan. Penelitian ini membahas penerapan algoritma Cosine Similarity dalam sistem matchmaking untuk mencocokkan pemain berdasarkan tingkat kesamaan atribut. Algoritma ini digunakan untuk mengukur kesamaan antar pemain dengan merepresentasikan data pemain sebagai vektor berdimensi tiga yang terdiri dari Damage Score, Skill Score, dan Synergy Score. Melalui implementasi menggunakan data pemain sintetik, hasil menunjukkan bahwa metode ini mampu menciptakan tim yang seimbang berdasarkan tingkat kesamaan atribut pemain. Studi ini memberikan wawasan baru dalam pengembangan sistem matchmaking yang lebih optimal dan dapat diterapkan pada berbagai genre game online.

Kata kunci*—Cosine Similarity, matchmaking, game online, skill-based matchmaking.

I. Pendahuluan

Perkembangan teknologi digital yang pesat telah mendorong terciptanya berbagai inovasi dalam industri hiburan, termasuk dalam dunia game online. Salah satu aspek yang sangat penting dalam pengalaman bermain game online adalah sistem matchmaking, yaitu mekanisme yang digunakan untuk mencocokkan pemain berdasarkan berbagai kriteria, seperti tingkat kemampuan, preferensi permainan, atau gaya bermain. Matchmaking yang efektif dapat meningkatkan kepuasan pemain, memperpanjang waktu bermain, dan menciptakan ekosistem permainan yang sehat.

Dalam implementasinya, berbagai metode digunakan untuk menyempurnakan sistem matchmaking. Salah satu metode yang menonjol adalah penerapan Cosine Similarity. Cosine Similarity merupakan algoritma berbasis vektor yang digunakan untuk mengukur kesamaan antara dua entitas. Dalam konteks game online, metode ini dapat digunakan untuk mencocokkan pemain berdasarkan profil atau data mereka, seperti preferensi karakter, skor permainan, atau pola perilaku.

Penerapan Cosine Similarity dalam sistem matchmaking memiliki keunggulan karena algoritma ini mampu menangani data multidimensi dan memberikan hasil yang akurat meskipun data memiliki skala yang berbeda. Dengan demikian, metode ini dapat membantu pengembang game menciptakan pengalaman bermain yang lebih adil dan menyenangkan bagi para pemain.

Makalah ini bertujuan untuk membahas implementasi Cosine Similarity dalam sistem matchmaking pada game online, termasuk konsep dasar algoritma, proses implementasi, dan evaluasi efektivitasnya dalam meningkatkan pengalaman bermain. Dengan penelitian ini, diharapkan dapat memberikan wawasan baru bagi pengembang game dalam mengoptimalkan sistem matchmaking untuk menciptakan lingkungan permainan yang lebih baik.

II. Dasar Teori

A. Matchmaking

Matchmaking adalah proses mencocokkan pemain dalam game online untuk membentuk tim atau menentukan lawan yang seimbang berdasarkan kriteria tertentu. Proses ini bertujuan untuk menciptakan pengalaman bermain yang kompetitif, adil, dan menyenangkan bagi semua pemain. Dalam konteks game online, matchmaking menjadi komponen penting karena berperan langsung dalam menjaga kepuasan pemain dan keberlanjutan komunitas permainan.

B. Skill-Based Matchmaking (SBMM)

Pada game yang bersifat lebih kompetitif, matchmaking yang diterapkan secara spesifik adalah skill-based matchmaking (SBMM), yaitu sistem pencocokan pemain berdasarkan tingkat kemampuan atau skill level. SBMM menggunakan metrik seperti skor, statistik permainan, atau peringkat untuk mencocokkan pemain dengan lawan dan rekan tim yang memiliki kemampuan serupa.

Pendekatan ini bertujuan untuk menciptakan permainan yang lebih seimbang, sehingga setiap pemain memiliki peluang yang sama untuk berkontribusi dan meraih kemenangan. Implementasi SBMM sangat relevan dalam game kompetitif karena mampu mengurangi ketimpangan kemampuan yang dapat mengganggu pengalaman bermain.

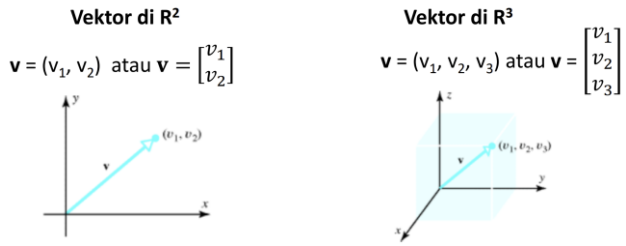
C. Vektor

Vektor adalah objek matematika yang direpresentasikan sebagai kumpulan elemen bernilai yang memiliki arah dan besar (magnitudo). Vektor dapat digunakan untuk merepresentasikan data atau atribut tertentu, seperti posisi, kecepatan, atau nilai-nilai lain yang dapat dibandingkan dalam konteks tertentu.

Secara umum, vektor dinyatakan sebagai deret angka terurut:

$$v = (x_1, x_2, x_3, \dots, x_n)$$

Dengan $x_1, x_2, x_3, \dots, x_n$ sebagai komponen vektor.



Gambar II.1 *Representasi vektor secara geometri di ruang dimensi dua dan ruang dimensi tiga*
Sumber: [1]

D. Perkalian Titik Vektor (Dot Product)

Perkalian titik (dot product) adalah operasi matematika antara dua vektor yang menghasilkan nilai skalar. Operasi ini digunakan untuk mengukur tingkat hubungan atau keselarasan antara dua vektor, terutama dalam konteks kesamaan arah.

Jika terdapat dua buah vektor u dan v, yang masing-masing dinyatakan sebagai:

$$u = (u_1, u_2, u_3, \dots, u_n), \qquad v = (v_1, v_2, v_3, \dots, v_n)$$

Maka perkalian titik $u \cdot v$ didefinisikan sebagai:

$$u \cdot v = u_1v_1 + u_2v_2 + u_3v_3 + \cdots + u_nv_n$$

Perkalian titik juga dapat dinyatakan menggunakan panjang (*magnitudo*) vektor dan sudut di antara keduanya:

$$u \cdot v = \|u\|\|v\| \cos\theta$$

Dengan:
- $\|u\|$ sebagai panjang vektor u.
- $\|u\|$ sebagai panjang vektor u.
- θ sebagai sudut antara kedua vektor.

Jika:
- $\cos\theta > 0$: Kedua vektor searah.
- $\cos\theta = 0$: Kedua vektor tegak lurus.
- $\cos\theta < 0$: Kedua vektor berlawanan arah.

E. Cosine Similarity

Dengan menyusun kembali persamaan yang telah dijelaskan sebelumnya, kita dapat menyatakan kesamaan dua buah vektor dengan lebih eksplisit:

$$sim(u, v) = \cos\theta = \frac{u \cdot v}{\|u\|\|v\|}$$

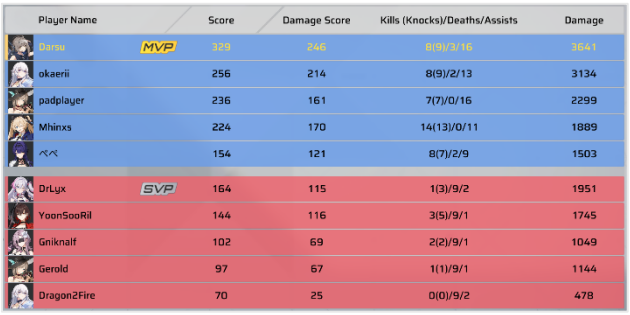
Seperti dengan persamaan sebelumnya, nilai kosinus akan berkisar antara -1 hingga 1, dengan nilai yang lebih mendekati 1 menyatakan dua vektor yang sangat mirip. Persamaan ini diterapkan dalam algoritma yang disebut dengan metode Cosine Similarity. Metode ini banyak digunakan dalam berbagai bidang, seperti analisis data, sistem rekomendasi, dan, dalam konteks makalah ini, sistem matchmaking pada game online.

III. ANALISIS

Kemampuan seorang pemain dapat diukur melalui berbagai parameter dan statistik yang mencerminkan keahlian, pengalaman, serta gaya bermainnya. Namun, parameter yang digunakan untuk mengukur kemampuan ini bervariasi tergantung pada jenis game yang dimainkan. Dalam konteks makalah ini, fokus pembahasan akan diarahkan pada genre hero shooter.

Pada genre shooter, elemen-elemen seperti jumlah eliminasi (kills), jumlah kematian (deaths), jumlah bantuan (assists), tingkat akurasi, serta aspek-aspek lainnya menjadi indikator utama untuk menilai performa seorang pemain.

Sebagai studi kasus, kita akan menganalisis elemen-elemen performa pemain pada game Strinova, sebuah game team-based hero shooter dengan mekanisme "skill" yang berbeda untuk setiap karakternya. Game ini menyediakan data statistik yang kaya dan terperinci untuk dievaluasi, mencakup berbagai parameter seperti efektivitas peran pemain, kontribusi terhadap tim, serta performa individual.



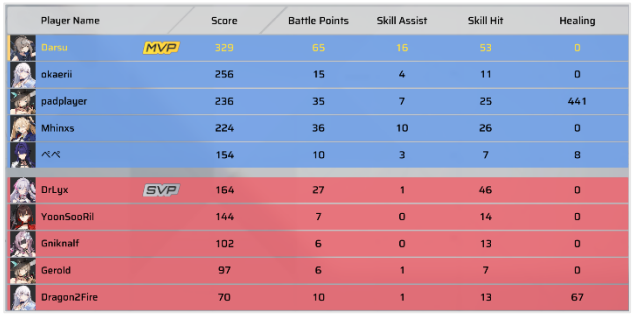
Player Name		Score	Damage Score	Kills (Knocks)/Deaths/Assists	Damage
Darsu	MVP	329	246	8(9)/3/16	3641
okaerii		256	214	8(9)/2/13	3134
padplayer		236	161	7(7)/0/16	2299
Mhinxs		224	170	14(13)/0/11	1889
^v^		154	121	8(7)/2/9	1503
DrLyx	SVP	164	115	1(3)/9/2	1951
YoonSooRil		144	116	3(5)/9/1	1745
Gniknalf		102	69	2(2)/9/1	1049
Gerold		97	67	1(1)/9/1	1144
Dragon2Fire		70	25	0(0)/9/2	478

Gambar III.1 *Statistik "Damage Score" pada match history Strinova*
Sumber: Dokumen Penulis

Pada gambar pertama, terdapat statistik "Damage Score" yang menggambarkan total kerusakan yang berhasil dihasilkan oleh pemain terhadap lawan selama pertandingan. Nilai kerusakan ini menjadi metrik penting karena lebih representatif dibandingkan hanya mengandalkan jumlah eliminasi (kills). Dalam banyak kasus, eliminasi tidak terjadi hanya dengan satu serangan langsung, melainkan hasil kontribusi dari beberapa pemain yang memberikan kerusakan secara bertahap.

Sebagai contoh, seorang pemain mungkin memberikan sebagian besar kerusakan pada lawan, tetapi eliminasi akhirnya dilakukan oleh pemain lain. Oleh karena itu, damage score memberikan gambaran yang lebih lengkap

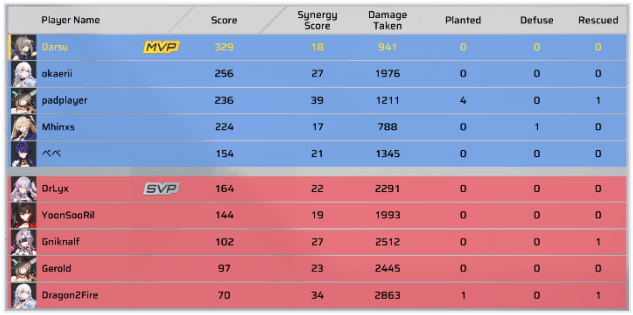
mengenai efektivitas pemain dalam menyerang, terlepas dari siapa yang mendapatkan kredit untuk eliminasi.



Gambar III.2 *Statistik "Skill" pada match history Strinova*
Sumber: Dokumen Penulis

Pada gambar kedua, data yang ditampilkan memberikan fokus pada berbagai aspek terkait keterampilan dalam penggunaan "Skill" pada karakter yang dimainkan. "Battle Points" merepresentasikan akumulasi dari berbagai elemen yang berhubungan dengan efektivitas penggunaan "Skill" karakter selama pertandingan. Statistik ini memberikan gambaran tentang seberapa efektif seorang pemain memanfaatkan "Skill" dari karakter yang dimainkan, sekaligus kontribusinya terhadap keberhasilan tim secara keseluruhan.

Dalam permainan berbasis tim, tidak hanya keterampilan individu yang penting, tetapi juga kemampuan pemain untuk berkolaborasi dan mendukung dinamika tim. Kerja sama yang baik antara pemain sering kali menjadi faktor penentu kemenangan, sehingga pemahaman dalam penggunaan "Skill" untuk berkontribusi terhadap permainan dalam konteks tim menjadi hal yang sangat krusial.



Gambar III.3 *Statistik "Synergy Score" pada match history Strinova*
Sumber: Dokumen Penulis

Pada gambar ketiga, terdapat statistik bernama "Synergy Score" yang memberikan penilaian terhadap seberapa baik seorang pemain berkontribusi dalam konteks kerja sama tim. "Synergy Score" mencerminkan elemen-elemen yang mendukung koordinasi tim, seperti penerimaan damage untuk tim (yang berarti pemain tersebut secara strategis menahan serangan lawan demi melindungi rekan satu tim atau menciptakan peluang bagi mereka), pemenuhan objektif permainan (menanam atau menjinakkan bom), serta menyelamatkan rekan tim yang telah "knocked".

Dengan beberapa statistik ini, kita dapat merancang suatu vektor pemain yang sederhana untuk diimplementasikan dalam sistem matchmaking berbasis skill. Dalam rangka menjaga kesederhanaan demonstrasi, statistik yang lebih kompleks seperti tingkat akurasi, "*win rate*", atau statistik detail lainnya tidak akan digunakan. Sebagai gantinya, vektor pemain akan dibangun berdasarkan tiga elemen utama, yaitu "Damage Score", "Skill Score" (disebut "Battle Points" pada gambar), dan "Synergy Score".

Ketiga elemen ini dipilih karena secara langsung merepresentasikan kemampuan ofensif, efektivitas penggunaan karakter, serta kontribusi terhadap kerja sama tim. "Damage Score" mencerminkan total kerusakan yang diberikan oleh pemain kepada lawan, "Skill Score" merepresentasikan efektivitas pemain dalam memanfaatkan kemampuan karakter mereka, dan "Synergy Score" menggambarkan kontribusi pemain dalam mendukung dinamika tim dan pencapaian objektif permainan.

Dengan vektor pemain ini, penerapan metode seperti Cosine Similarity dapat dilakukan untuk membandingkan pemain berdasarkan tingkat keterampilan mereka. Pendekatan ini memungkinkan perancangan sistem matchmaking yang lebih seimbang dan adil, memastikan pemain dipasangkan dengan rekan dan lawan yang sesuai dengan kemampuan mereka.

IV. IMPLEMENTASI

Implementasi sistem matchmaking berbasis Cosine Similarity dikembangkan menggunakan bahasa Python. Untuk keperluan demonstrasi, data pemain dibuat secara sintetik guna menyimulasikan atribut yang relevan dalam sistem matchmaking. Proses implementasi ini terdiri dari tiga tahap utama: pembuatan data pemain, representasi vektor pemain, dan pencocokan pemain berdasarkan tingkat kesamaan atribut yang dihitung.

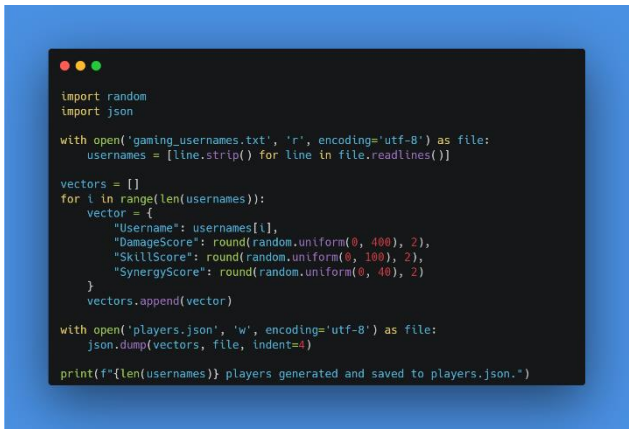
A. Implementasi Pembuatan Data Pemain

Pertama, nama-nama untuk pemain dibuat menggunakan skrip generateNames.py. Skrip ini menghasilkan 1000 nama pengguna unik dengan berbagai pola, seperti kombinasi kata sifat, kata benda, istilah permainan, dan karakter khusus. Nama-nama yang dihasilkan disimpan dalam file gaming_usernames.txt.

```
import random

def generate_gaming_usernames(count=1000):
    adjectives = ['Dark', 'Epic', 'Shadow', 'Dragon', 'Cyber', 'Neon', 'Frost', 'Storm', 'Ninja',
                  'Ghost', 'Elite', 'Royal', 'Silent', 'Swift', 'Mystic', 'Flame', 'Toxic', 'Thunder',
                  'Pixel', 'Chaos']

    nouns = ['Slayer', 'Knight', 'Hunter', 'Warrior', 'Master', 'Legend', 'Phoenix', 'Demon', 'Wolf',
             'Assassin', 'Guardian', 'Striker', 'Sniper', 'Reaper', 'King', 'Beast', 'Hawk', 'Dragon',
             'Phantom', 'Scout']

    gaming_terms = ['Pro', 'Gaming', 'Player', 'Boss', 'MVP', 'Elite', 'Ace', 'Champion', 'Prime',
                    'Master']

    usernames = set()

    while len(usernames) < count:
        pattern = random.randint(1, 4)

        if pattern == 1:
            username = f"{random.choice(adjectives)}{random.choice(nouns)}{random.randint(0, 999)}"
        elif pattern == 2:
            username = f"{random.choice(gaming_terms)}{random.choice(adjectives)}{random.randint(0, 99)}"
        elif pattern == 3:
            username = f"{random.choice(nouns)}{random.choice(gaming_terms)}{random.randint(0, 999)}"
        else:
            special_chars = ['x', '_', '-']
            char = random.choice(special_chars)
            username = f"{char}{random.choice(nouns)}{char}"

        usernames.add(username)

    return list(usernames)

usernames = generate_gaming_usernames(1000)

with open('gaming_usernames.txt', 'w') as f:
    for username in usernames:
        f.write(username + '\n')

print(f"Successfully generated {len(usernames)} unique usernames and saved them to
'gaming_usernames.txt'")
```

Gambar IV.1 *Implementasi pembuatan nama pemain*
Sumber: Dokumen Penulis

Selanjutnya, nama-nama tersebut akan dibaca oleh skrip generatePlayers.py yang kemudian menghasilkan atribut pemain berupa Damage Score, Skill Score, dan Synergy Score. Data pemain yang telah dibuat disimpan dalam format JSON di file players.json untuk mempermudah pengolahan dan analisis pada tahap implementasi berikutnya.



```
import random
import json

with open('gaming_usernames.txt', 'r', encoding='utf-8') as file:
    usernames = [line.strip() for line in file.readlines()]

vectors = []
for i in range(len(usernames)):
    vector = {
        "Username": usernames[i],
        "DamageScore": round(random.uniform(0, 400), 2),
        "SkillScore": round(random.uniform(0, 100), 2),
        "SynergyScore": round(random.uniform(0, 40), 2)
    }
    vectors.append(vector)

with open('players.json', 'w', encoding='utf-8') as file:
    json.dump(vectors, file, indent=4)

print(f"{len(usernames)} players generated and saved to players.json.")
```

Gambar IV.2 *Implementasi pembuatan data pemain*
Sumber: Dokumen Penulis

B. Implementasi Representasi Vektor Pemain

Setiap pemain direpresentasikan sebagai vektor berdimensi tiga yang terdiri dari:

- *Damage Score*: Menggambarkan kemampuan ofensif pemain.
- *Skill Score*: Mengukur efektivitas penggunaan karakter dalam permainan.
- *Synergy Score*: Menilai kontribusi pemain terhadap kerja sama tim.

Pada skrip matchmaking.py, fungsi read_players() berfungsi untuk membaca data pemain dari file JSON, sementara fungsi vector() mengonversi data suatu pemain menjadi format vektor numerik yang siap digunakan dalam proses perhitungan kesamaan.

```
import json

def read_players():
    with open('players.json', 'r', encoding='utf-8') as file:
        return json.load(file)

def vector(player):
    return [player['DamageScore'], player['SkillScore'], player['SynergyScore']]
```

Gambar IV.3 *Implementasi representasi vektor pemain*
Sumber: Dokumen Penulis

C. Implementasi Pencocokan Pemain

Pencocokan pemain dilakukan dengan menggunakan perhitungan Cosine Similarity untuk mengukur tingkat kesamaan antara vektor pemain. Fungsi cosine_similarity() bertugas menghitung nilai kesamaan antara dua vektor pemain, sementara fungsi matchmake() menentukan pasangan pemain terbaik dengan memeringkat skor kesamaan gaya bermain (playstyle similarity) serta memperhitungkan kesenjangan keterampilan (skill gap) untuk memastikan kecocokan yang optimal.

```
import numpy as np

def cosine_similarity(a, b):
    a = np.array(a)
    b = np.array(b)
    return np.dot(a, b) / (np.linalg.norm(a) * np.linalg.norm(b))

def calculate_score(player1, player2):
    playstyle_similarity = cosine_similarity(vector(player1), vector(player2))
    skill_gap = abs(np.linalg.norm(vector(player1)) - np.linalg.norm(vector(player2)))
    score = playstyle_similarity / (skill_gap if skill_gap > 1 else 1)
    return score

def matchmake(players, user, n):
    scores = []
    for player in players:
        if player['Username'] != user['Username']:
            score = calculate_score(user, player)
            scores.append((player['Username'], score))
    scores.sort(key=lambda x: -x[1])
    return scores[:n]
```

Gambar IV.4 *Implementasi pencocokan pemain*
Sumber: Dokumen Penulis

Pencocokan ini diintegrasikan lebih lanjut dalam fungsi main(), yang menjalankan seluruh alur sistem matchmaking. Fungsi ini memulai proses dengan membaca data pemain dari file JSON menggunakan read_players(), kemudian memilih seorang pemain utama secara acak untuk dijadikan pusat pencocokan. Pemain-pemain lainnya dibandingkan dengan pemain utama menggunakan fungsi matchmake(), yang menghasilkan daftar pemain terbaik berdasarkan skor kesamaan (similarity score).

Hasil pencocokan kemudian digunakan untuk membentuk dua tim: tim sekutu dan tim lawan. Dalam pembentukan tim, pemain-pemain dari daftar hasil pencocokan disusun secara bergantian untuk menciptakan kedua tim. Pemain utama dimasukkan ke tim sekutu terlebih dahulu, diikuti oleh pemain pertama dari daftar ke tim lawan, pemain kedua ke tim sekutu, pemain ketiga ke tim lawan, dan seterusnya. Pendekatan bergantian ini memastikan distribusi pemain yang seimbang di antara kedua tim berdasarkan skor kesamaan mereka dengan pemain utama, sehingga tidak hanya mencocokkan pemain secara adil tetapi juga memberikan hasil yang transparan dan terukur.

Untuk memberikan konteks yang lebih luas, fungsi get_ratings() digunakan untuk menghitung dan menampilkan distribusi kemampuan pemain dalam dataset. Statistik seperti nilai minimum, maksimum, dan rata-rata norma vektor pemain disajikan untuk memberikan gambaran menyeluruh tentang tingkat keterampilan keseluruhan pemain.

```python
import random
import numpy as np

def get_ratings(players):
    norms = [np.linalg.norm(vector(p)) for p in players]
    min_norm = min(norms).round(2)
    max_norm = max(norms).round(2)
    avg_norm = (sum(norms) / len(norms)).__round__(2)

    print(f"(Min: {min_norm}, Max: {max_norm}, Avg: {avg_norm})\n")

def main():
    players = read_players()
    user = players[int(random.uniform(0, len(players)))]
    matches = matchmake(players, user, 9)
    print(f"Match for {user['Username']} with rating
{np.linalg.norm(vector(user)).round(2)}:")
    get_ratings(players)

    print("Ally Team:")
    print(user['Username'])
    for i in range(1, 9, 2):
        print(f"{matches[i][0]} ({matches[i][1].round(2)})")

    print("\nEnemy Team:")
    for i in range(0, 9, 2):
        print(f"{matches[i][0]} ({matches[i][1].round(2)})")

if __name__ == '__main__':
    main()
```

Gambar IV.5 *Implementasi fungsi utama*
Sumber: Dokumen Penulis

V. HASIL DAN PEMBAHASAN

Pengujian sistem matchmaking yang diimplementasikan dilakukan menggunakan data pemain sintetik yang dihasilkan. Data ini mencakup atribut untuk 1000 pemain yang unik. Proses pengujian ini bertujuan untuk mengevaluasi efektivitas sistem dalam mencocokkan pemain secara adil dan seimbang berdasarkan tingkat kesamaan atribut mereka.

Tahap pertama dalam pengujian adalah generasi data pemain menggunakan skrip generateNames.py dan generatePlayers.py. Hasil dari proses ini adalah file players.json, yang berisi data pemain dalam format JSON. Setiap entri dalam file tersebut mencakup nama pengguna dan nilai untuk ketiga atribut utama. File ini menjadi dasar bagi sistem matchmaking, memungkinkan data untuk dibaca, diproses, dan dianalisis pada tahap implementasi selanjutnya.

```json
[
    {
        "Username": "SniperChampion214",
        "DamageScore": 210.01,
        "SkillScore": 24.85,
        "SynergyScore": 22.79
    },
    {
        "Username": "ScoutGaming444",
        "DamageScore": 347.74,
        "SkillScore": 88.22,
        "SynergyScore": 20.23
    },
    {
        "Username": "xSlayerx",
        "DamageScore": 87.28,
        "SkillScore": 47.13,
        "SynergyScore": 33.31
    },
    {
        "Username": "GuardianAce101",
        "DamageScore": 98.48,
        "SkillScore": 18.72,
        "SynergyScore": 10.11
    },
  ...
]
```

Gambar V.1 *Cuplikan players.json*
Sumber: Dokumen Penulis

Setelah data pemain berhasil dibuat, tahap berikutnya adalah menjalankan proses matchmaking dengan memilih seorang pemain secara acak sebagai pusat pencocokan. Hasil dari proses matchmaking ini akan dianalisis untuk mengevaluasi seberapa efektif sistem dalam membentuk tim yang seimbang. Analisis akan mencakup pemeriksaan tingkat kesamaan dalam setiap tim, hubungan tingkat kesamaan dengan distribusi keterampilan berdasarkan norma vektor pemain, serta pemeriksaan kecocokan antara dua pemain secara manual melalui players.json.

A. Hasil Uji Percobaan 1

```
Match for AceFrost76 with rating 148.91:
(Min: 10.88, Max: 410.46, Avg: 209.09)

Ally Team:
AceFrost76
MVPDark15 (0.99)
KnightBoss703 (0.98)
MasterPlayer572 (0.69)
AceFrost42 (0.65)

Enemy Team:
GuardianPlayer718 (1.0)
MVPDark73 (0.99)
HawkMaster691 (0.95)
MasterPrime29 (0.69)
MVPRoyal71 (0.61)
```

Tabel V.1 *Output uji percobaan 1*
Sumber: Dokumen Penulis

Pada percobaan pertama, terlihat bahwa pembentukan tim dalam pertandingan memiliki tingkat kesamaan yang cukup baik. Hubungan dengan distribusi keterampilan tidak akan dibahas dalam percobaan ini untuk saat ini. Sebagai gantinya, kita akan menggali lebih dalam mengenai kecocokan antara dua pemain, yaitu AceFrost76 dan MVPDark15.

```
{
    "Username": "AceFrost76",
    "DamageScore": 115.66,
    "SkillScore": 92.81,
    "SynergyScore": 13.49
},
{
    "Username": "MVPDark15",
    "DamageScore": 104.92,
    "SkillScore": 98.55,
    "SynergyScore": 34.86
}
```


Tabel V.2 *AceFrost76 dan MVPDark15 pada players.json*
Sumber: Dokumen Penulis

Dapat dilihat bahwa nilai-nilai antara kedua pemain, AceFrost76 dan MVPDark15, sangat mirip sehingga secara langsung berkontribusi pada tingkat kesamaan yang tinggi dalam pembentukan tim. Hal ini membuktikan bahwa algoritma yang digunakan mampu secara efektif mengelompokkan pemain dengan kemampuan yang serupa sehingga mendukung terciptanya tim yang seimbang dan pertandingan yang kompetitif.

B. Hasil Uji Percobaan 2

```
Match for -Warrior- with rating 28.86:
(Min: 10.88, Max: 410.46, Avg: 209.09)

Ally Team:
-Warrior-
DragonPlayer327 (0.78)
DarkDragon946 (0.23)
GhostAssassin882 (0.17)
AssassinPrime217 (0.1)

Enemy Team:
ReaperGaming293 (0.87)
LegendPlayer13 (0.71)
BossStorm43 (0.18)
PhoenixChampion636 (0.12)
ThunderGuardian329 (0.09)
```

Tabel V.3 *Output uji percobaan 2*
Sumber: Dokumen Penulis

Pada percobaan kedua, pemain utama dipilih dengan rating yang berada jauh di bawah median rata-rata pemain. Hal ini menyebabkan pencocokan dengan pemain lain memiliki tingkat kesamaan yang rendah secara keseluruhan. Sesuai dengan distribusi normal, jumlah pemain yang berada dalam rentang kecocokan dengan pemain utama menjadi jauh lebih sedikit. Akibatnya, algoritma mengalami kesulitan dalam menciptakan tim yang seimbang ketika terdapat perbedaan ekstrem dalam kemampuan pemain.

C. Pembahasan

potensi sebagai metode yang efektif untuk mengelompokkan pemain berdasarkan tingkat kesamaan kemampuan. Namun, algoritma ini masih memiliki beberapa kekurangan, terutama ketika terdapat perbedaan ekstrem dalam kemampuan pemain. Kekurangan ini dapat dikaitkan dengan terbatasnya jumlah parameter yang digunakan untuk perbandingan antar pemain. Idealnya, kemampuan seorang pemain diukur melalui parameter yang lebih beragam, seperti akurasi, *winrate*, respons waktu, dan lain sebagainya.

Selain itu, penting untuk memasukkan statistik negatif dalam algoritma, karena implementasi saat ini hanya menggunakan statistik positif seperti “damage score”, “skill score”, dan “synergy score”. Dengan memperhitungkan statistik negatif, seperti jumlah kematian, *team damage*, atau aspek lainnya, algoritma akan memiliki gambaran yang lebih lengkap tentang kemampuan seorang pemain. Penggunaan parameter yang lebih beragam dan seimbang ini diharapkan dapat meningkatkan akurasi algoritma serta menghasilkan pencocokan matchmaking yang lebih adil dan kompetitif.

VI. KESIMPULAN DAN SARAN

Implementasi Cosine Similarity dalam sistem matchmaking pada game online terbukti mampu menjadi solusi yang efektif dalam menciptakan pengalaman bermain yang lebih adil dan seimbang. Hasil pengujian menunjukkan bahwa sistem mampu mengelompokkan pemain dengan kemampuan yang serupa, menghasilkan tim yang kompetitif dalam beberapa skenario.

Namun, sistem ini memiliki beberapa keterbatasan yang perlu diatasi untuk meningkatkan akurasinya. Salah satu kelemahannya adalah terbatasnya jumlah parameter yang digunakan yang dapat memengaruhi hasil pencocokan dalam kasus perbedaan ekstrem pada kemampuan pemain. Selain itu, sistem ini hanya menggunakan statistik positif, sehingga belum mampu memberikan gambaran yang sepenuhnya menyeluruh tentang kemampuan pemain.

Untuk mengatasi keterbatasan tersebut, disarankan agar sistem memperluas parameter yang digunakan, termasuk akurasi tembakan, rasio eliminasi dan kematian, waktu bermain, atau respons waktu, sehingga mampu memberikan representasi kemampuan pemain yang lebih komprehensif. Selain itu, memasukkan statistik negatif seperti jumlah kematian atau *team damage* dapat menyeimbangkan evaluasi pemain dan menciptakan hasil pencocokan yang lebih adil.

Dengan penyempurnaan algoritma yang mempertimbangkan dinamika tim, gaya bermain, atau preferensi mode permainan, sistem ini berpotensi memberikan pengalaman bermain yang lebih optimal bagi para pemain serta mendukung ekosistem game yang lebih kompetitif dan sehat.

VII. LAMPIRAN

Program yang digunakan dalam implementasi makalah ini dapat diakses melalui repositori GitHub berikut:
https://github.com/Darsua/Cosine-Sim-Matchmaking

VIII. UCAPAN TERIMA KASIH

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga makalah yang berjudul “Implementasi Cosine Similarity dalam Sistem Matchmaking pada Game Online” dapat diselesaikan dengan baik tanpa hambatan yang berarti.

Penulis juga mengucapkan terima kasih kepada dosen pengampu mata kuliah IF2123 Aljabar Linier dan Geometri, terutama kepada Bapak Dr. Rila Mandala, Bapak Dr. Rinaldi Munir, Bapak Dr. Judhi Santoso, serta Bapak Arrival Dwi Sentosa, M.T, atas ilmu, bimbingan, dan arahan yang telah diberikan selama proses penyusunan makalah ini. Segala dukungan yang diterima sangat berkontribusi dalam menyempurnakan isi dan kualitas makalah ini.

REFERENSI

[1] Munir, Rinaldi. (2023). "Aljabar Geometri (Bagian 1)". https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-27-Aljabar-Geometri-Bagian1-2023.pdf

[2] GameTree. What Is Matchmaking In Games?. Diakses dari https://gametree.me/gaming-terms/matchmaking/

[3] NetDuma. How Matchmaking Works. Diakses dari https://netduma.com/blog/how-matchmaking-works

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 27 Desember 2024

Darrel Adinarya Sunanda
13523061