Integrasi Fuzzy Logic dan Weight-Graph Representation pada Weighted Hybrid MBTI-Based Job Recommender System

Moh Fairuz Alauddin Yahya - 13522057
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13522057@itb.ac.id

***Abstract*—Keputusan mengenai prospek karir seringkali memunculkan kebingungan dan ketakutan dalam menentukan arah karir. Penempatan individu dalam karier yang sejalan dengan kepribadian mereka menjadi tantangan krusial. Sebagai solusi, Myers-Briggs Type Indicator (MBTI) muncul sebagai alat populer untuk mengukur dan mengklasifikasikan kepribadian manusia, namun penggunaannya membawa beberapa keterbatasan yang menimbulkan ketidakpastian. Dalam rangka mengatasi ketidakpastian hasil tes MBTI dan kaitannya dengan karier, makalah ini mengusulkan implementasi sistem rekomendasi pekerjaan berbasis MBTI yang menggabungkan Logika Fuzzy dan Representasi Graf. Sistem ini juga memanfaatkan Content-Based Filtering (CBF) dan Collaboration Filtering (CF) yang tergabung dalam Hybrid Recommender System. Dengan pendekatan ini, sistem rekomendasi dapat memberikan pandangan yang lebih akurat dan mendalam tentang kekuatan, kelemahan, dan gaya kerja individu. Sebagai hasilnya, individu dapat membuat keputusan karier yang lebih terinformasi dan tepat sesuai dengan karakteristik unik mereka.**

***Keywords*—Graf, *Hybrid Recommender System, Fuzzy Logic*, MBTI, Pekerjaan.**

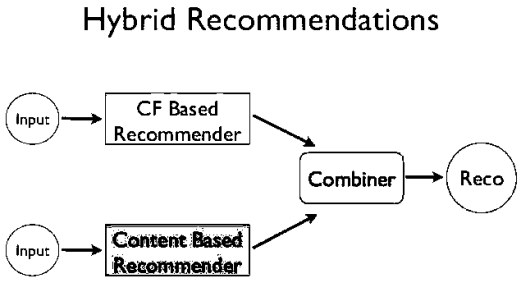
I. PENDAHULUAN

Dalam era dinamika lapangan kerja yang terus berkembang dan penuh tantangan, penempatan individu dalam karier yang sesuai dengan kepribadian mereka menjadi sebuah aspek krusial. Keputusan mengenai prospek karir seringkali menjadi langkah yang menentukan bagi banyak orang, dan ketidakpastian ini dapat menyebabkan kebingungan serta ketakutan dalam menentukan arah karir yang ingin mereka tempuh. Sejalan dengan kompleksitas ini, muncul kebutuhan akan alat yang dapat memberikan pandangan yang mendalam tentang kepribadian seseorang untuk memudahkan pengambilan keputusan terkait karir.

Dalam konteks ini, Myers-Briggs Type Indicator (MBTI) muncul sebagai salah satu alat yang populer dan banyak digunakan untuk mengukur dan mengklasifikasikan kepribadian manusia. MBTI didasarkan pada teori psikologis Carl Jung, yang mengidentifikasi empat dimensi utama yang memengaruhi perilaku manusia. Penggunaan MBTI tidak hanya memberikan pemahaman mendalam tentang kekuatan dan kelemahan individu, tetapi juga menyediakan wawasan tentang gaya kerja mereka, yang merupakan fondasi penting dalam memilih pekerjaan yang sesuai.

Namun, perlu dicatat bahwa MBTI, meskipun sangat berguna, tidak bersifat mutlak dalam menentukan kepribadian seseorang, sehingga tidak jarang saat seseorang melakukan tes MBTI hasilnya bisa berbeda tergantung klasifikasi dan banyaknya pertanyaan. Inilah yang menjadi tantangan utama, dimana kebanyakan dari aplikasi tes tersebut hanya mengklasifikasikan MBTI berdasarkan *scale* pertanyaan yang diajukan saja.

Oleh karena itu, makalah ini memperkenalkan konsep sistem sistem rekomendasi pekerjaan berbasis MBTI yang mengintegrasikan penyaringan berbasis konten dan kolaboratif menggabungkan *Fuzzy Logic* dan Representasi Graf. Makalah ini membahas implementasi *Fuzzy Logic* untuk normalisasi skala kepribadian, perhitungan bobot untuk setiap skala, dan pembentukan profil pekerjaan dengan mempertimbangkan *fuzzy ogic* dan penggunaan *Content Based Filtering (CBF)*, *Collaboration Filtering (CF*) yang dibungkus menjadi *Hybrid Recommender System*.



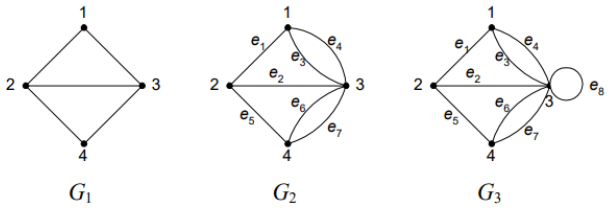
***Gambar 1.1.** Hybrid Recommender System, diambil dari* [1]

II. LANDASAN TEORI

A. Graf

Graf dapat didefinisikan sebagai struktur diskrit yang terdiri dari kumpulan simpul (vertices) yang saling terhubung melalui himpunan sisi (edges) dengan keterkaitan tertentu. Representasi graf ini dapat dituliskan dalam bentuk $G = (V, E)$, di mana G adalah graf, V merupakan himpunan simpul-simpul $v_1, v_2, \dots, v_n$ yang tidak kosong, dan E adalah himpunan sisi-sisi $e_1, e_2, \dots, e_n$ yang menghubungkan sepasang simpul dalam graf.

Dalam konteks graf, sepasang simpul dapat dihubungkan oleh dua sisi yang berbeda, yang dikenal sebagai sisi ganda (multiple edges). Selain itu, terdapat sisi yang berawal dan berakhir pada simpul yang sama, yang disebut sebagai gelang atau kalang (loop). Berdasarkan keberadaan sisi ganda atau sisi gelang, graf dapat diklasifikasikan menjadi graf sederhana, yang tidak memiliki sisi ganda dan sisi gelang, serta graf tak-sederhana, yang mengandung sisi ganda atau gelang, dan jika graf tak-sederhana memiliki gelang, disebut sebagai graf-semu.

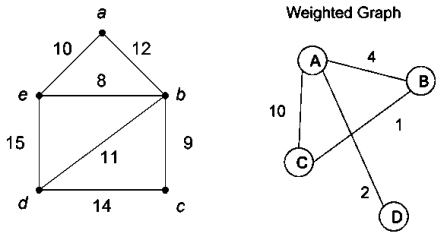


Gambar 2.1. *(a) Graf sederhana (b) Graf tak-sederhana (c) Graf-semu (Sumber :* [2]*)*

Orientasi arah sisi juga menjadi faktor penting dalam mengelompokkan graf. Graf tak berarah memiliki sisinya tanpa orientasi arah, sementara graf berarah memiliki orientasi arah pada setiap sisinya.

Beberapa terminologi dalam teori graf meliputi:

1. Ketetanggan (adjacency): Dua simpul dalam graf dikatakan bertetangga jika terhubung oleh setidaknya satu sisi.
2. Bersisian (incidency): Sebuah sisi dikatakan bersisian dengan dua simpul tertentu jika menghubungkannya.
3. Derajat (degree): Jumlah sisi yang bersisian dengan suatu simpul.
4. Lintasan (path): Barisan simpul dan sisi yang menghubungkannya dari simpul awal ke simpul tujuan.
5. Siklus (cycle atau circuit): Lintasan yang berawal dan berakhir pada simpul yang sama.
6. Keterhubungan (connected): Dua simpul dikatakan terhubung jika terdapat lintasan yang menghubungkannya.
7. Graf berbobot (*weight*ed graph): Graf yang setiap sisinya memiliki nilai atau bobot spesifik.

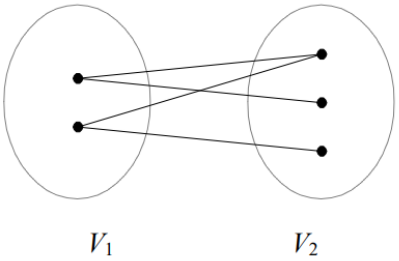


Gambar 2.2. *Contoh orientasi graf berbobot (Sumber :* [2]*)*

8. Graf lengkap (complete graph): Graf yang setiap simpulnya terhubung dengan semua simpul lainnya dalam graf tersebut.

Beberapa jenis graf khusus, antara lain:

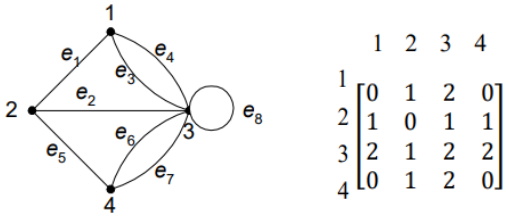
1. Graf Lengkap (Complete Graph)
   Graf lengkap adalah jenis graf sederhana di mana setiap simpulnya terhubung dengan semua simpul lainnya, sehingga jumlah sisi pada graf dengan n simpul selalu $\frac{n(n-1)}{2}$. Graf lengkap yang memiliki n simpul dinotasikan sebagai Kn.
2. Graf Lingkaran
   Graf lingkaran adalah graf sederhana di mana setiap simpulnya memiliki derajat dua.
3. Graf Teratur (Regular Graph)
   Graf teratur adalah jenis graf yang setiap simpul memiliki derajat yang sama, sehingga jumlah sisi pada graf teratur adalah $nr/2$, di mana n adalah jumlah simpul dan r adalah derajat setiap simpul. Graf teratur berderajat r adalah graf di mana semua simpul memiliki derajat r.
4. Graf Bipartite (Bipartite Graph)
   Graf bipartite adalah graf G yang himpunan simpulnya dapat dibagi menjadi dua bagian, V_1 dan V_2, sehingga setiap sisi dalam G menghubungkan simpul dari V_1 ke simpul dari V_2. Graf bipartite direpresentasikan sebagai $G(V_1, V_2)$.



Gambar 2.3. *Orientasi graf bipartite (Sumber :* [2]*)*

Beberapa representasi graf adalah sebagai berikut:

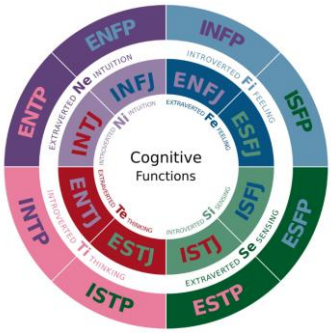
1. Matriks Ketetanggaan (Adjacency Matrix)
   Sebuah graf tak berarah dengan n simpul dapat diwakili oleh matriks ketetanggaan M $n \, x \, n$. Elemen pada baris ke-i dan kolom ke-j bernilai 1 jika terdapat sisi yang menghubungkan simpul ke-i dan simpul ke-j, dan 0 jika tidak. Pada graf berarah, elemen pada baris ke-i dan kolom ke-j akan bernilai 1 jika ada sisi yang mengarah dari simpul ke-i ke simpul ke-j. Jika graf memiliki bobot pada sisinya, elemen pada posisi yang sama menggambarkan bobot sisi yang menghubungkan simpul ke-i dan simpul ke-j.
2. Matriks Bersisian (Incidency Matrix)
   Graf tak berarah dengan n simpul dan m sisi dapat direpresentasikan oleh matriks bersisian M $n \, x \, m$. Elemen pada baris ke-i dan kolom ke-j bernilai 1 jika simpul ke-i bersisian dengan sisi ke-j, dan 0 jika tidak.



Gambar 2.4. *Orientasi graf bipartite (Sumber :* [2]*)*

B. MBTI

MBTI, singkatan dari *Myers-Briggs Type Indicator*, adalah alat sederhana yang membantu dalam memahami perbedaan dalam kepribadian [3]. Dikembangkan berdasarkan teori Carl Jung, MBTI mengelompokkan orang ke dalam salah satu dari enam belas tipe kepribadian berdasarkan preferensi mereka dalam empat dimensi. Masing-masing dimensi ini memiliki dua pilihan yang berlawanan, membentuk 16 tipe kepribadian yang berbeda, yaitu *Extraversion vs. Introversion*, *Sensing vs. Intuition*, *Thinking vs. Feeling*, dan *Judging vs. Perceiving*, membentuk dasar bagi pemahaman tentang bagaimana mengambil energi, mengumpulkan informasi, membuat keputusan, dan berurusan dengan dunia luar. Dalam dunia kerja, MBTI digunakan untuk mengembangkan tim yang efektif, memahami perbedaan kepemimpinan, dan meningkatkan manajemen konflik, sehingga memberikan kontribusi positif terhadap produktivitas dan kesejahteraan di lingkungan kerja.



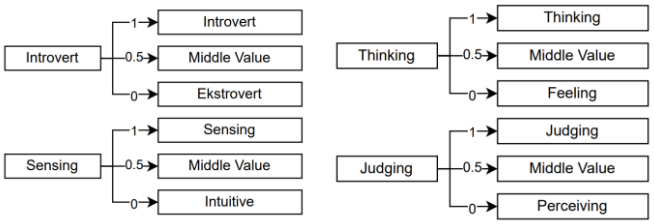
Gambar 2.5. *16 Kepribadian MBTI (Sumber:* [4]*)*

C. Fuzzy Logic

Fuzzy Logic adalah suatu metode penyelesaian masalah yang mampu mengatasi ketidakpastian dan kompleksitas dengan memperkenalkan konsep keanggotaan yang tidak bersifat biner (dua nilai, yaitu satu dan nol), melainkan dapat mengambil nilai di antara dua keadaan tersebut. Logika ini digunakan untuk menafsirkan dan mengelola besaran yang diekspresikan dalam bahasa alami, memungkinkan komputer untuk mengatasi informasi yang tidak pasti atau ambigu [5].

Fuzzy Logic memanfaatkan konsep variabel linguistik dan aturan-aturan yang bersifat *fuzzy* atau kabur untuk menggambarkan pengetahuan manusia. Konsep keanggotaan *fuzzy* memperkenalkan kemungkinan bahwa suatu elemen dapat sebagian termasuk dalam satu kategori dan sebagian lagi dalam kategori lain, dengan tingkat keanggotaan yang dapat berkisar antara 0 dan 1.

Salah satu kegunaan *Fuzzy Logic* adalah dalam normalisasi nilai. Normalisasi adalah proses mengubah nilai-nilai variabel menjadi rentang tertentu agar dapat diproses atau dibandingkan dengan mudah. Dalam konteks ini, *Fuzzy Logic* dapat digunakan untuk menyesuaikan dan menormalisasi nilai-nilai yang tidak pasti atau kabur. Sebagai contoh, *Fuzzy Logic* juga dapat diterapkan dalam menentukan kepribadian introvert atau ekstrovert. Dengan memanfaatkan variabel linguistik dan aturan yang bersifat kabur, *Fuzzy Logic* dapat menggambarkan kompleksitas sifat manusia yang seringkali sulit diukur secara pasti.



Gambar 2.6. *Desain sistem fuzzy logic yang diajukan (Sumber: Dokumentasi Pribadi)*

D. Recommender System

Sistem Rekomendasi, atau yang dikenal sebagai *Recommender Systems*, merupakan suatu alat dan teknik perangkat lunak yang bertujuan memberikan rekomendasi kepada pengguna mengenai produk atau konten yang mungkin bermanfaat atau menarik bagi mereka. Tujuannya adalah memberikan rekomendasi berdasarkan preferensi personal pengguna serta menggunakan informasi riwayat mereka terkait produk atau konten yang telah dilihat dan disukai. Seiring perkembangan teknologi, ada beberapa pendekatan yang umum digunakan dalam Sistem Rekomendasi, termasuk *Content-Based Filtering*, *Collaborative Filtering*, dan kombinasi atau *hybrid recommender*.

1. *Content-Based Filtering (CBF)*

   CBF menggunakan informasi tentang karakteristik dan konten suatu item untuk memberikan rekomendasi kepada pengguna. Dalam konteks MBTI, misalnya, *CBF* akan mengevaluasi hasil *scale* yang telah dinormalisasi dengan *fuzzy logic* yang terkait dengan karakteristik MBTI yang dimiliki oleh pengguna.

   CBF dapat dijelaskan dengan persamaan matematis yang melibatkan hasil skala yang telah dinormalisasi dengan *fuzzy logic* yang terkait dengan karakteristik MBTI pengguna. Sebagai contoh, jika S adalah hasil skala yang dinormalisasi dan F adalah fungsi *fuzzy logic* yang merepresentasikan karakteristik MBTI tertentu, maka persamaan dapat dirumuskan sebagai berikut:

$$CBF(S,F) = \sum_{i=1}^{n} S_i \cdot F_i$$

   Dengan menganalisis kesamaan karakteristik antara hasil perhitungan *user profile* tersebut dengan data *fuzzy scale* pekerjaan baru atau lama, *CBF* dapat memberikan rekomendasi yang sesuai dengan preferensi individual. Keuntungan utama dari *CBF* adalah kemampuannya untuk memberikan rekomendasi untuk item yang baru atau tidak populer berdasarkan karakteristik yang sesuai dengan pengguna.

2. *Collaborative Filtering (CF)*

   CF memanfaatkan informasi dari sekelompok pengguna untuk memberikan rekomendasi. *User-Based CF* mengidentifikasi pengguna dengan preferensi serupa dengan pengguna yang sedang direkomendasikan. *Item-Based CF*, di sisi lain, merekomendasikan item berdasarkan kesamaan karakteristik dengan item yang disukai pengguna. Dalam konteks *user-based* dapat digambarkan persamaan matematisnya sebagai berikut:

$$CF_{User}(u,i) = \frac{\sum_{v \in N(u)} Sim(u,v) \cdot R(v,i)}{\sum_{v \in N(u)} | Sim(u,v) |}$$

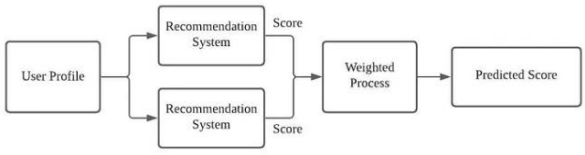
Dimana $CF_{User}(u,i)$ adalah prediksi rekomendasi untuk pengguna u terhadap item i, $N(u)$ adalah himpunan pengguna yang mirip dengan $u, Sim(u,v)$ adalah nilai kesamaan antara pengguna u dan pengguna v, dan $R(v,i)$ adalah peringkat yang diberikan pengguna v untuk item i.

Kelebihan utama *CF* adalah kemampuannya untuk memberikan rekomendasi berdasarkan pola perilaku dan preferensi yang mungkin tidak terlihat dalam karakteristik konten saja. Namun, tantangan utama adalah cold start problem, yaitu kesulitan memberikan rekomendasi untuk item baru atau pengguna baru.

3. *Hybrid Recommender*

Sistem Rekomendasi Hibrida menggabungkan pendekatan *CBF* dan *CF* untuk memaksimalkan kelebihan masing-masing metode. Dengan menyatukan informasi dari karakteristik konten dan pola perilaku pengguna, sistem hibrida dapat menghasilkan rekomendasi yang lebih akurat dan personal.

Pendekatan sistem rekomendasi hybrid menawarkan berbagai strategi, termasuk pendekatan *weighted* yang menggabungkan hasil model dengan penimbangan statis, pendekatan *switching* yang memilih model berdasarkan situasi, pendekatan *mixed* yang menggunakan berbagai set dataset kandidat, pendekatan *feature combination* yang menambahkan model kontributor virtual, pendekatan *feature augmentation* yang meningkatkan kinerja sistem inti, pendekatan *cascade* yang menyelesaikan masalah kecil pada hasil utama, dan pendekatan *meta-level* yang menggantikan dataset asli dengan model dari kontributor. Setiap pendekatan memiliki keunikannya sendiri, memberikan fleksibilitas dalam menghadapi tantangan rekomendasi yang beragam [6].



Gambar 2.7. *Alur Weighted Hybrid Recommender System (Sumber:* [7])

III. METODOLOGI

A. Batasan Masalah

Terdapat beberapa batasan yang digunakan sebagai langkah penyederhanaan masalah. Batasan-batasan tersebut terletak pada data yang didapatkan, yaitu:

1. Data hanya berisi skala tes kuesioner yang terbatas hanya pada 142 pertanyaan, memfokuskan pada variabel-variabel tertentu untuk menghindari kompleksitas yang berlebihan.
2. Data bidang pekerjaan hanya terdiri dari 12 bidang pekerjaan berdasarkan responden pengisi kuesioner dari dapat yang didapatkan.

B. Data yang digunakan

Data yang digunakan berasal dari dataset *Kaggle*. Dataset ini merujuk pada kumpulan pertanyaan dari kuesioner dengan markup kepuasan responden terhadap pekerjaan mereka. Background data ini merupakan versi modifikasi dari MBTI, yaitu kuesioner laporan diri introspektif yang menunjukkan preferensi psikologis berbeda dalam cara seseorang mempersepsikan dunia dan mengambil keputusan. Dataset ini mencakup 21.846 jawaban kuesioner yang dikumpulkan secara online oleh *KeyHabits*, dan setiap rekaman mengandung jawaban terhadap 142 pertanyaan kuesioner, nilai mentah skala, psikotipe yang dihitung, dan jawaban terhadap pertanyaan dalam bentuk *json*.

	jobtitle	jobfield	scale_e	scale_i	scale_s	scale_n	scale_t	scale_f	scale_j	scale_p	psychotype	sati
1	HR manager	Staff and training	16	22	17	16	23	11	22	15	ISTJ	
2	HR manager	Staff and training	28	6	14	18	20	16	20	15	ENTJ	
3	HR manager	Staff and training	28	10	22	10	16	14	22	12	ESTJ	
4	HR manager	Staff and training	24	10	16	17	13	23	31	3	ENFJ	
5	HR manager	Staff and training	25	12	17	13	23	11	10	25	ESTP	
...	...	...	...	...	...	...	...	...	...	...	...	
12617	Lawyer	Specialists	19	16	19	15	16	17	15	17	ESFP	
12618	Lawyer	Specialists	5	32	25	9	11	23	29	8	ISFJ	
12619	Lawyer	Specialists	25	14	15	16	18	22	9	25	ENFP	
12620	Lawyer	Specialists	21	16	27	9	30	8	32	2	ESTJ	
12621	Lawyer	Specialists	15	18	20	9	28	6	33	6	ISTJ	

12621 rows × 12 columns

Gambar 3.1. *Cuplikan data (Sumber : Dokumentasi Pribadi).*

C. Pemodelan Masalah

Diambil beberapa sample data yang dimiliki oleh MBTI tertentu, misalnya data yang dimiliki beberapa orang yang memiliki MBTI ENTJ sebagai berikut

Nama Pekerjaan	***Job Count***
Staff and training	40
IT	120
Specialist	34

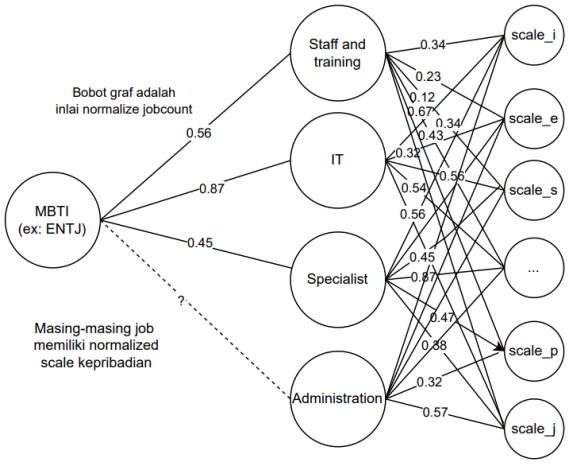
Tabel 3.1. *Sampel data pekerjaan MBTI ENTJ (input user)*

Dari informasi tersebut diperlukan informasi yang yang lebih kecil untuk lebih dilakukan kalkulasi dan perbandingan data lebih detail. Untuk itu, perlu diekstraksi item-item yang ada pada setiap pekerjaan. Hal tersebut dapat dimudahkan dengan membuat suatu matrix seperti berikut

Nama Pekerjaan	***scale* i**	***scale* e**	***scale* s**	***scale* n**	**..**
Staff and training	28	12	40	20	..
IT	21	19	21	39	..
Specialist	17	23	12	48	..

Tabel 3.2. *Sampel scale sub-pekerjaan MBTI ENTJ*

Dari data input user dan *scale* kepribadian yang didapatkan kemudian dapat dimodelkan sebagai graf berbobot dengan setiap sisi mengandung nilai dari hasil normalisasi *scale* kepribadian. Berikut adalah pemodelan permasalahan sebagai graf berbobot (bobot diisi tidak seluruhnya karena tumpang tindih).



***Gambar 3.2**. Pemodelan permasalahan sebagai graf bipartite berbobot yang telah dinormalisasi (Sumber : Dokumentasi Pribadi).*

D. Perhitungan dan Implementasi Program

Berdasarkan landasan teori pada bab sebelumnya, penulis membuat implementasi algoritma *Hybrid Recommender System* yang memanfaatkan *Content Based Filtering* dan *Collaborative Filtering User-Based* dengan pendekatan pemrograman dinamis menggunakan bahasa pemrograman Python. Implementasi ini dimodifikasi dan dibuat dalam masing-masing *class* sesuai algoritma yang digunakan.

Inisialisasi import `numpy` dan `sklearn` untuk melakukan perhitungan, data cleansing dan load *kaggle* data.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import NearestNeighbors
from sklearn.preprocessing import MinMaxScaler
import seaborn as sns
from pandas.plotting import table


# Set maximum number of rows and columns displayed
pd.set_option('display.max_rows', 100)
pd.set_option('display.max_columns', 100)
```

0.0s

***Gambar 3.3.** Kode impor modul program utama (Sumber : Dokumentasi Pribadi).*

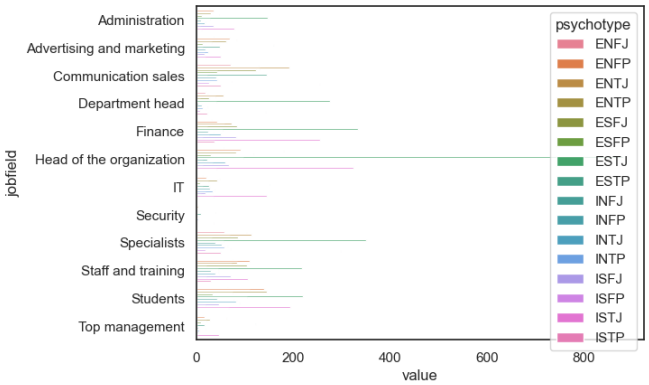
```python
# Load data
df = pd.read_csv('data/kpmi_data.csv')

# Data cleansing
df = df.dropna()

# Drop unused columns analysis
for i in range(1, 143):
    try:
        df.drop('q'+str(i), axis=1, inplace=True)
    except KeyError:
        print(f"Kolom 'q{i}' tidak ada di dataframe.")

# Show initial data
df
```

***Gambar 3.4.** Data cleansing (Sumber : Dokumentasi Pribadi).*



***Gambar 3.5.** Data hubungan pekerjaan dengan psychotype (Sumber : Dokumentasi Pribadi).*

a. Content Based Filtering (CBF)

Implementasi algoritma *CBF* diawali dengan melakukan normalisasi setiap *scale* kepribadian dan jumlah pekerjaan dalam 1 *psychotype* menggunakan *scaler* pada fungsi `get_fuzzy_scale`. Kemudian perlu kalikan hasil normalisasi tersebut untuk membentuk matriks *weighted scale* pada fungsi `get_weighted_scale`.

```python
class CBFRecommender:
    def __init__(self, df):
        self.df = df
        self.job_counts = df.groupby('jobfield')['psychotype'].value_counts()
        self.scaler = MinMaxScaler()

    def get_fuzzy_scale(self, scale):
        return self.scaler.fit_transform(self.df[[scale]])

    def get_weighted_scale(self, scale):
        # Convert job_counts to a dictionary
        job_counts_dict = self.job_counts.to_dict()

        # Map the job_counts to the DataFrame
        job_counts_mapped = self.df.set_index(['jobfield', 'psychotype'])\
            .index.map(job_counts_dict)

        # Normalize the mapped job_counts
        normalized_job_counts = self.scaler.fit_transform(job_counts_mapped
                                                          .values.reshape(-1,1)).flatten()

        # Multiply the normalized scale with the normalized job_counts
        weighted_scale = self.df['normalized_'+scale] * normalized_job_counts

        return weighted_scale
```

***Gambar 3.6.** Algoritma Content Based Filtering 1 (Sumber : Dokumentasi Pribadi).*

Dari *weighted scale* perlu dicari *job profile* masing-masing *psychotype* menggunakan fungsi `build_psychotype_profile`. Lalu akan dicari

pekerjaan yang direkomendasikan berdasarkan profil kepribadian yang sudah didapatkan terurut berdasarkan skor rekomendasi yang terbesar.

```
def build_psychotype_profile(self):
    job_profile = self.df.groupby('psychotype')[['weighted_scale_e',
        'weighted_scale_i', 'weighted_scale_s', 'weighted_scale_n',
        'weighted_scale_t', 'weighted_scale_f', 'weighted_scale_j',
        'weighted_scale_p']].sum()
    scaler = MinMaxScaler()
    normalized_job_profile = pd.DataFrame(scaler.fit_transform(job_profile),
                                          columns=job_profile.columns,
                                          index=job_profile.index)
    return normalized_job_profile

def recommend(self, input_psychotype):
    # Get unique user profile
    unique_job_profile = self.get_unique_psychotype_profile(self.build_psychotype_profile())

    # Calculate recommendation scores for new jobs
    for psychotype in self.df['psychotype'].unique():
        self.df.loc[self.df['psychotype'] == psychotype, 'cbf_score'] = sum(
            self.df.loc[self.df['psychotype'] == psychotype, 'normalized_'+scale] *
            unique_job_profile[input_psychotype]['job_profile_'+scale]
            for scale in ['scale_e', 'scale_i', 'scale_s', 'scale_n', 'scale_t',
                          'scale_f', 'scale_j', 'scale_p']
        )

    # Sort by recommendation score and select only 'jobfield' and 'cbf_score'
    recommendations = self.df.sort_values(by='cbf_score', ascending=False)\
        [['jobtitle','jobfield', 'cbf_score']]

    return recommendations
```

Gambar 3.7. *Algoritma Content Based Filtering 2 (Sumber : Dokumentasi Pribadi).*

Semua pemrosesan algoritma *CBF* tertuang dan diatur dalam fungsi `process`.

```
def get_unique_job_profile(self, normalized_job_profile):
    return {
        psychotype:{
            'job_profile_'+scale: normalized_job_profile.loc[psychotype]['weighted_'+scale]
            for scale in ['scale_e', 'scale_i', 'scale_s', 'scale_n',
                          'scale_t', 'scale_f', 'scale_j', 'scale_p']
        } for psychotype in normalized_job_profile.index.unique()
    }

def process(self):
    # Normalize each scale
    for scale in ['scale_e', 'scale_i', 'scale_s', 'scale_n', 'scale_t',
                  'scale_f', 'scale_j', 'scale_p']:
        self.df['normalized_'+scale] = self.get_fuzzy_scale(scale)

    # Calculate weights for each scale based on the number of jobs in the same field
    for scale in ['scale_e', 'scale_i', 'scale_s', 'scale_n', 'scale_t',
                  'scale_f', 'scale_j', 'scale_p']:
        self.df['weighted_'+scale] = self.get_weighted_scale(scale)

    # Form user profile by summing scales in the weight matrix for each movie
    normalized_job_profile = self.build_job_profile()

    # Add job profile to the dataframe
    self.df = self.df.join(normalized_job_profile, on='psychotype', rsuffix='_job')

    # Get unique user profile
    unique_job_profile = self.get_unique_job_profile(normalized_job_profile)

    return self.df, unique_job_profile
```

Gambar 3.8. *Algoritma Content Based Filtering 2 (Sumber : Dokumentasi Pribadi).*

b. Collaborative Filtering (CF)

Prinsip *Collaborative Filtering (CF) user based* melibatkan faktorisasi matriks awal, diikuti pengisian nilai kosong dengan prediksi menggunakan algoritma *nearest neighbors* dengan metrik *cosine*. Hasil *similarity index* dan *matriks jobcount subset* yang telah dinormalisasi dijumlahkan untuk membentuk *weighted matrix*. Selanjutnya, psychotype yang berkaitan dengan pekerjaan yang sama dikumpulkan, dan skor rekomendasi dihitung dengan membagi jumlah *weighted matrix* dengan jumlah *similarity index*. Proses ini semuanya dibungkus dalam fungsi `recommend`.

```
class CFRecommender:
    def __init__(self, df):
        self.df = df
        self.job_counts = df.groupby(['psychotype', 'jobfield']).size()\
            .reset_index(name='counts')
        scaler = MinMaxScaler()
        self.job_counts['normalized_counts'] = scaler.fit_transform(
            self.job_counts['counts'].values.reshape(-1,1)).flatten()
        self.matrix = self.job_counts.pivot(index='psychotype',
                columns='jobfield', values='normalized_counts').fillna(0)
        self.model_knn = NearestNeighbors(metric='cosine',
                                          algorithm='brute')
        self.model_knn.fit(self.matrix)

    def recommend(self, input_psychotype, n_neighbors=16):
        distances, indices = self.model_knn.kneighbors(self.matrix.
            loc[input_psychotype].values.reshape(1, -1), n_neighbors=n_neighbors)
        weighted_job_counts = self.matrix.copy()
        for i in range(len(distances.flatten())):
            weighted_job_counts.iloc[i] *= distances.flatten()[i]
        recommendation_values = weighted_job_counts.sum() / distances.sum()
        recommendations = recommendation_values.sort_values(ascending=False)
        recommendations_df = pd.DataFrame(recommendations).reset_index()
        recommendations_df.columns = ['jobfield', 'cf_score']

        # Add jobtitle to the recommendations
        recommendations_df = recommendations_df.merge(self.df[['jobfield', 'jobtitle']].
                                drop_duplicates(), on='jobfield', how='left')

        # Drop duplicates based on jobfield
        recommendations_df = recommendations_df.drop_duplicates(subset='jobfield',
                                                                keep='first')

        # Reorder the columns
        recommendations_df = recommendations_df[['jobfield', 'jobtitle', 'cf_score']]

        return recommendations_df
```

Gambar 3.9. *Algoritma Collaborative Filtering (Sumber : Dokumentasi Pribadi).*

c. Hybrid Recommender System

Implementasi dari algoritma *weighted hybrid recommender system* cukup dengan mengkombinasikan hasil rekomendasi dari algoritma *CF* dan *CBF* dengan porsi *weight* masing-masing. Disini penulis menyamakan *weight*nya yaitu bernilai 1 dengan menjumlahkan rekomendasi keduanya.

```
class HybridRecommender:
    def __init__(self, df, cf_weight=1, cbf_weight=1):
        self.df = df
        self.cf_recommender = CFRecommender(df)
        self.cbf_recommender = CBFRecommender(df)
        self.cf_weight = cf_weight
        self.cbf_weight = cbf_weight

    def recommend(self, input_psychotype, n_neighbors=16):
        # Get recommendations from both models
        cf_recommendations = self.cf_recommender.recommend(input_psychotype, n_neighbors)
        cbf_recommendations = self.cbf_recommender.recommend(input_psychotype)

        # Now you can merge the DataFrames
        combined_recommendations = pd.merge(cf_recommendations, cbf_recommendations,
                                            on='jobfield', how='outer').fillna(0)
        # Calculate the hybrid score based on the weights
        combined_recommendations['hybrid_score'] = (combined_recommendations['cf_score']
                    * self.cf_weight) + (combined_recommendations['cbf_score']
                                         * self.cbf_weight)

        # Sort by the hybrid score

        combined_recommendations = combined_recommendations.drop_duplicates(
            subset='jobfield', keep="first").reset_index().sort_values(by='hybrid_score',
                                                                       ascending=False)

        return combined_recommendations
```

Gambar 3.10. *Algoritma Weighted Hybrid Recommender (Sumber : Dokumentasi Pribadi).*

IV. ANALISIS DAN PEMBAHASAN

A. Hasil Eksekusi Program

a. Content Based Filtering (CBF)

Hasil eksekusi program pada *CBF* untuk merekomendasikan top 5 pekerjaan berdasarkan MBTI ENTP adalah

```
# Initialize the CBFRecommender
cbfrecommender = CBFRecommender(df)

# Process the dataframe and get unique user profile
df, unique_job_profile = cbfrecommender.process()
#  Calculate similarity scores and get top 5 recommendations
recommendations = cbfrecommender.recommend('ENTP')
top_5_recommendations = recommendations.head(5).reset_index(drop=True)
top_5_recommendations
```

✓ 0.4s

	jobtitle	jobfield	cbf_score
0	Head (marketing advertising)	Advertising and marketing	0.232587
1	System administrator	IT	0.232470
2	Student	Students	0.232043
3	Journalist	Specialists	0.230219
4	Teacher	Staff and training	0.229579

Gambar 4.1. *Implementasi recommender system menggunakan CBF (Sumber : Dokumentasi Pribadi).*

Terlihat bahwa 5 bidang pekerjaan yang direkomendasikan untuk ENTP dengan metode *CBF* adalah *advertising and marketing, IT, students, specialist,* dan *staff and training*. ENTP adalah seorang ahli debat yang suka mengambil peran oposisi. Penulis merasa hasil yang didapatkan sudah cukup relevan dengan skor rekomendasi disekitar angka 0,23.

Penulis juga melakukan testing dengan memasukkan job baru yang tidak ada sebelumnya pada data untuk melihat skor rekomendasi atas suatu job berdasarkan *scale* sebagai berikut

```
scales = [('scale_e', 'scale_i'), ('scale_s', 'scale_n'), ('scale_t', 'scale_f'), ('scale_j'
# Buat objek DataFrame dengan skala menggunakan loop
new_jobs = pd.DataFrame({
    'jobtitle': ['Job Title 1', 'Job Title 2', 'Job Title 3'],
    'jobfield': ['New Job Field 1', 'New Job Field 2', 'New Job Field 3'],
    'psychotype': ['ISFJ', 'ISFJ', 'ISFJ'],
})
for scale1, scale2 in scales:
    new_jobs[scale1] = np.random.rand(3)
    new_jobs[scale2] = 1 - new_jobs[scale1]
for psychotype in new_jobs['psychotype'].unique():
    new_jobs.loc[new_jobs['psychotype'] == psychotype,
                 'recommendation_score'] = sum(
        new_jobs.loc[new_jobs['psychotype'] == psychotype, scale] *
        unique_job_profile[psychotype]['job_profile_'+scale]
        for scale in ['scale_e', 'scale_i', 'scale_s', 'scale_n',
                      'scale_t', 'scale_f', 'scale_j', 'scale_p'])
new_jobs.sort_values(by='recommendation_score', ascending=False, inplace=True)

new_jobs = new_jobs[['recommendation_score', 'jobtitle', 'jobfield', 'psychotype', 'scale_e'

# Display the DataFrame
new_jobs
```

0.0s

	recommendation_score	jobtitle	jobfield	psychotype	scale_e	scale_i	scale_s	scale_n	scale_t	sc
	0.083593	Job Title 3	New Job Field 3	ISFJ	0.056642	0.943358	0.520678	0.479322	0.307111	0.69
	0.069915	Job Title 1	New Job Field 1	ISFJ	0.498921	0.501079	0.843533	0.156467	0.495073	0.50
	0.066669	Job Title 2	New Job Field 2	ISFJ	0.730280	0.269720	0.510808	0.489192	0.436586	0.56

Gambar 4.2. *Implementasi recommender system menggunakan CBF 2 (Sumber : Dokumentasi Pribadi).*

Hasil eksekusi kode tersebut terlihat bahwa skor rekomendasi berkisar diangka 0,6-0,8 terurut membuktikan bahwa seorang pengguna tertentu bisa berkonsultasi terkait kecocokan terhadap suatu pekerjaan berdasarkan *scale* psyschotype.

b. Collaborative Filtering (CF)

Hasil eksekusi program pada *CF* untuk merekomendasikan top 5 pekerjaan berdasarkan MBTI ENTP adalah

```
# Initialize the recommender
recommender = CFRecommender(df)

# Get recommendations for a specific psychotype
recommendations = recommender.recommend('ENTP')
top_5_recommendations = recommendations.head(5).reset_index(drop=True)
top_5_recommendations
```

0.0s

```
\Tubes\Matdis\venv\Lib\site-packages\sklearn\base.py:465: UserWarning: X does not
warnings.warn(
```

	jobfield	jobtitle	cf_score
	Head of the organization	Businessman	0.140649
	Communication sales	Sales Agent (real estate insurance)	0.134964
	Specialists	Sales Agent (real estate insurance)	0.109980
	Students	Graduate student	0.101652
	Finance	Financial analyst	0.091209

Gambar 4.3. *Implementasi recommender system menggunakan CF (Sumber : Dokumentasi Pribadi).*

Terlihat bahwa 5 bidang pekerjaan yang direkomendasikan untuk ENTP dengan metode *CF* adalah *head of the organizer, communication sales, specialists, students, dan finance.* Penulis merasa bahwa bidang pekerjaan tersebut masih relevan dengan seorang ENTP yang suka mengambil peran oposisi. Penulis merasa hasil yang didapatkan sudah cukup relevan dengan skor rekomendasi disekitar angka 0,09-0,14.

c. Hybrid Recommender System

Hasil eksekusi program pada *weight hybrid recommender* untuk merekomendasikan top 5 pekerjaan berdasarkan MBTI ENTP adalah

```
# Initialize the recommender
hybrid_recommender = HybridRecommender(df)

Get recommendations for a specific psychotype
recommendations = hybrid_recommender.recommend('ENTP')
top_5_recommendations = recommendations.head(5)
top_5_recommendations
```

0.2s

```
\Tubes\Matdis\venv\Lib\site-packages\sklearn\base.py:465: UserWarning: X does not have valid feature name
warnings.warn(
```

	index	jobfield	jobtitle_x	cf_score	jobtitle_y	cbf_score	hybrid_score
	0	Head of the organization	Businessman	0.140649	General director	0.224915	0.365564
	1	Communication sales	Sales Agent (real estate insurance)	0.134964	Account Manager	0.227965	0.362928
	2	Specialists	Sales Agent (real estate insurance)	0.109980	Journalist	0.230219	0.340199
	3	Students	Graduate student	0.101652	Student	0.232043	0.333695
	4	Finance	Financial analyst	0.091209	Head (Finance Law)	0.213251	0.304460

Gambar 4.4. *Implementasi recommender system menggunakan weight hybrid (Sumber : Dokumentasi Pribadi).*

Terlihat bahwa 5 bidang pekerjaan yang direkomendasikan untuk ENTP dengan metode *hybrid* adalah kombinasi skor dari metode *CF* dan *CBF*, yaitu *head of the organizer, communication sales, specialists, students, dan finance* dengan skor rekomendasi berkisar 0,3 – 0,36.

V. KESIMPUAN

Integrasi *Fuzzy Logic* dan *Weight-Graph Representation* pada *Hybrid MBTI-Based Job Recommender System* memberikan kontribusi positif terhadap kemampuan sistem rekomendasi pekerjaan. Hasil eksekusi program menunjukkan bahwa metode hybrid ini, yang menggabungkan keunggulan *Fuzzy Logic* dan *Weight-Graph Representation*, mampu memberikan rekomendasi pekerjaan dengan tingkat akurasi dan relevansi

yang baik dengan kepribadian secara nyata. *Fuzzy Logic* digunakan untuk menangani ketidakpastian dan kompleksitas dalam menilai kesesuaian antara tipe kepribadian MBTI dan pekerjaan, sementara *Weight-Graph Representation* memberikan bobot pada hubungan antar elemen, meningkatkan keakuratan rekomendasi. Dengan demikian, pendekatan ini dapat dianggap sebagai langkah maju dalam menghadirkan solusi rekomendasi pekerjaan yang lebih canggih dan adaptif berdasarkan karakteristik psikologis pengguna.

VI. Ucapan Terima Kasih

Penulis menyampaikan ucapan terima kasih kepada:

1. Tuhan Yang Maha Esa atas berkat-Nya sehingga penulis dapat menyelesaikan makalah ini dengan baik,
2. Kedua orang tua penulis yang telah memberikan dukungan kepada penulis,
3. Bapak Dr. Ir. Rinaldi Munir, M.T., Ibu Fariska Zakhralativa Ruskanda S.T.,M.T, dan Ibu Dr. Nur Ulfa Maulidevi S.T., M.Sc selaku dosen mata kuliah IF2120 Matematika Diskrit yang telah membimbing penulis dan memberikan banyak ilmu yang bermanfaat selama perkuliahan

Referensi

[1] dataaspirant, "An Introduction to Recommendation Engines," 13 March 2015. [Online]. Available: https://dataconomy.com/2015/03/13/an-introduction-to-recommendation-engines/. [Accessed 11 December 2023].

[2] R. Munir, "https://informatika.stei.itb.ac.id/," [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/. [Accessed 11 December 2023].

[3] "MBTI," 29 September 2019. [Online]. Available: https://itp.psikologi.ui.ac.id/2019/09/29/mbti/. [Accessed 11 December 2023].

[4] "Ns Deveopment," [Online]. Available: https://nsd.co.id/posts/ketahui-tipe-kepribadian-seseorang-dengan-tes-mbti.html. [Accessed 11 December 2023].

[5] L. Zadeh, "*Fuzzy Logic*," in *Fuzzy Logic*, IEEE, 1998, pp. 83-93.

[6] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *springer*, p. 331–370, 2002.

[7] J. Chiang, "https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-*system*-3e4f78266ad8," Medium, [Online]. Available: https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-*system*-3e4f78266ad8. [Accessed 11 12 2023].

[8] "Machine Learning," Google Developers, 27 September 27. [Online]. Available: https://developers.google.com/machine-learning/recommendation/content-based/basics?hl=id. [Accessed 11 December 2023].

[9] R. Adam, "Pendekatan Durkheimian: Agama dalam Fungsi Sosialnya," 6 April 2021. [Online]. Available: https://crcs.ugm.ac.id/pendekatan-durkheimian-agama-dalam-fungsi-sosialnya/. [Accessed 11 December 2023].

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, 11 Desember 2023

Moh Fairuz Alauddin Yahya 13522057