Sistem Pemberi Rekomendasi Anime Menggunakan Pendekatan Hybrid

Tugas Akhir

diajukan untuk memenuhi salah satu syarat

memperoleh gelar sarjana

dari Program Studi Informatika

Fakultas Informatika

Universitas Telkom

1301184255

Alif Ranadian Nadhifah

Program Studi Sarjana Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2024

LEMBAR PENGESAHAN

Sistem Pemberi Rekomendasi Anime Menggunakan Pendekatan Hybrid

Anime Recommender System Using Hybrid Approach

NIM: 1301184255

Alif Ranadian Nadhifah

Tugas akhir ini telah diterima dan disahkan untuk memenuhi sebagian syarat memperoleh gelar pada Program Studi Sarjana Informatika
Fakultas Informatika
Universitas Telkom

Bandung, 09 Agustus 2024
Menyetujui

Pembimbing I	Pembimbing II
Agung Toto Wibowo, S.T, M.T., Ph.D.	-
NIP: 06810035	NIP:

Ketua Program Studi
Sarjana Informatika,

Dr. Erwin Budi Setiawan, S.Si., M.T.
NIP: 00760045

LEMBAR PERNYATAAN

Dengan ini saya, Alif Ranadian Nadhifah, menyatakan sesungguhnya bahwa Tugas Akhir saya dengan judul ”**Sistem Pemberi Rekomendasi Anime Menggunakan Pendekatan Hybrid**” beserta dengan seluruh isinya adalah merupakan hasil karya sendiri, dan saya tidak melakukan penjiplakan yang tidak sesuai dengan etika keilmuan yang belaku dalam masyarakat keilmuan. Saya siap menanggung resiko/sanksi yang diberikan jika dikemudian hari ditemukan pelanggaran terhadap etika keilmuan dalam buku TA atau jika ada klaim dari pihak lain terhadap keaslian karya.

Bandung, 09 Agustus 2024

Yang Menyatakan,

Alif Ranadian Nadhifah

Sistem Pemberi Rekomendasi Anime Menggunakan Pendekatan Hybrid

Alif Ranadian Nadhifah[1], Agung Toto Wibowo[2]

[1,2]Fakultas Informatika, Universitas Telkom, Bandung
[1]alifnadhifah@students.telkomuniversity.ac.id, [2]agungtoto@telkomuniversity.ac.id,

Abstrak

Sistem pemberi rekomendasi adalah alat dan perangkat software yang dapat memberikan solusi berupa saran suatu item yang bisa digunakan oleh user untuk mengambil sebuah keputusan untuk memilih item yang dikehendaki contohnya seperti memilih item anime. Sistem pemberi rekomendasi terbagi dalam beberapa kelas pendekatan rekomendasi yang berbeda diantaranya ada: collaborative filtering, content-based, dan hybrid recommender system. Namun perlu diketahui bahwa masing-masing kelas pendekatan tersebut memiliki kelemahan masing-masing. Sehingga penelitian pembangunan sistem pemberi rekomendasi anime ini memilih pendekatan hybrid filtering dengan tujuan mengatasi kelemahan dari masing-masing pendekatan yang berdiri sendiri. Penelitian ini terbatas hanya menggunakan 3 filtering untuk dikombinasikan, yaitu k-means clustering, TF-IDF dengan cosine similarity, dan item-based filtering dengan algoritma SVD. Dataset yang digunakan dalam penelitian ini merupakan dataset anime yang diambil melalui situs web kaggle yang sebelumnya telah dilakukan proses crawling dari situs myanimelist.net yang dibagikan secara open source. Akhir dari penelitian ini menemukan bahwa meskipun di beberapa penelitian lain pendekatan hybrid dapat meningkatkan kualitas rekomendasi ternyata sebaliknya. Masih ditemukan masalah-masalah seperti sparsity dan scalability dalam penelitian ini yang menyebabkan rekomendasi yang dihasilkan sistem masih kurang baik.

Kata kunci : Sistem Pemberi Rekomendasi, K-Means Clustering, TF-IDF, Item-based Filtering, SVD

Abstract

Recommendation systems are software and devices that can provide solutions in the form of item suggestions that can be used by users to make decisions about choosing the desired item, for example choosing an anime item. Comprehensive recommendation systems in several different recommendation approach classes include: Collaborative Filtering, Content-Based, and Hybrid Recommendation System. However, it should be noted that each of these approach classes has its own weaknesses. So this anime recommendation system development research chooses a hybrid filtering approach with the aim of overcoming the weaknesses of each stand-alone approach. This study is limited to using only 3 filters that will be combined, namely k-means clustering, TF-IDF with cosine similarity, and item-based filtering with the SVD algorithm. The dataset used in this study is an anime dataset taken from the kaggle website which has previously been crawled from the myanimelist.net site which is shared as open source. The end of this study found that although in several other studies the hybrid approach can improve the quality of recommendations, the opposite actually happened. Problems such as sparsity and scalability were still found in this study, which caused the recommendations produced by the system to be less than optimal.

Keywords: Recommender System, K-Means Clustering, TF-IDF, Item-based Filtering, SVD

1. Pendahuluan

Latar Belakang

Dunia hiburan saat ini berkembang pesat seimbang dengan berkembangnya teknologi. Bahkan dari waktu ke waktu juga bermunculan banyak aplikasi hiburan seperti aplikasi *streaming online*. Ada banyak hal yang ditawarkan dalam berbagai aplikasi *streaming online*, salah satunya adalah anime. Anime adalah salah satu hiburan yang yang berasal dari Jepang yang dapat dinikmati oleh seorang peminat anime yang bisa dikatakan sebagai seorang pengguna aplikasi tersebut. Dari tahun ke tahun, ada banyak anime yang diproduksi tiap tahunnya. Bagi pengguna yang menyukai menonton anime saat melihat fenomena ini pasti akan membuat mereka kebingungan dalam menentukan anime apa yang akan dipilih untuk ditontonnya. Maka dari itu, sistem pemberi rekomendasi dapat dijadikan sebagai pilihannya.

Sistem pemberi rekomendasi adalah alat dan perangkat *software* yang dapat memberikan solusi berupa saran suatu *item* yang bisa digunakan oleh *user* untuk mengambil sebuah keputusan untuk memilih *item* yang dikehendaki. Sistem pemberi rekomendasi beberapa kelas pendekatan rekomendasi yang berbeda diantaranya ada: *collaborative filtering*, *content-based*, dan *hybrid recommender system* [11][10].

Namun perlu diketahui bahwa masing-masing kelas pendekatan tersebut memiliki kelemahan masing-masing. Misalnya dalam pendekatan *collaborative filtering* yang paling populer dalam sistem pemberi rekomendasi memiliki masalah yang umum ditemukan, seperti *cold start*, *sparsity* dan *scalability* [5]. Untuk itu, pada penelitian ini dibangun sistem pemberi rekomendasi anime menggunakan pendekatan *hybrid filtering* yang mengkombinasikan beberapa pendekatan rekomendasi sebagai *filtering* rekomendasi anime. Apakah dengan dipilihnya pendekatan *hybrid filtering* dapat bisa meminimalisir ditemukannya masalah umum pada salah satu pendekatan *filtering* yang digunakan. Disamping itu, dataset yang digunakan dalam penelitian ini merupakan dataset anime yang didapat dari situs *web* kaggle yang sebelumnya telah dilakukan proses *crawling* dari situs myanimelist.net yang dibagikan secara *open source*.

Topik dan Batasannya

Penelitian ini dilakukan dengan mengimplementasikan pendekatan *hybrid filtering* pada sistem pemberi rekomendasi anime. Namun pendekatan *hybrid filtering* yang diterapkan terbatas dengan hanya menggunakan 3 filtering diantaranya *k-means clustering*, TF-IDF dengan *cosine similarity*, serta *item based collaborative filtering* dengan algoritma SVD.

Adapun dataset anime yang dimiliki terbagi dalam 3 data, diantara data *item* anime sebanyak 16216 item, data profil *user* sebanyak 47885 *user*, dan data ulasan *rating* sebanyak 130519 ulasan. Namun, dalam penelitian ini dataset anime yang digunakan hanya sebanyak 20% dari data *item* anime, 20% dari data profil *user*, dan data ulasan *rating* menyesuaikan dari data *item* anime dan data profil *user* yang tersisa. Pengurangan jumlah *dataset* ini adalah agar proses yang dibutuhkan untuk membangun sistem tidak memakan banyak waktu dan menghindari masalah *scalability*.

Tujuan

Penelitian ini bertujuan untuk mengimplementasikan pendekatan *hybrid filtering* pada sistem pemberi rekomendasi anime untuk bisa melihat apakah dengan melakukan kombinasi berbagai pendekatan *filtering* bisa meminimalisir ditemukannya masalah umum pada salah satu pendekatan *filtering* yang digunakan. Adapun hasil penelitian dilihat melalui performansi akurasi terhadap sistem yang dibangun menggunakan metrik evaluasi, seperti *recall*, *precission*, dan *f1-score*.

Organisasi Tulisan

Struktur penulisan jurnal penelitian setelah sub-bab pendahuluan akan dilanjutkan dengan pembahasan studi terkait pada sub-bab kedua. Kemudian pada sub-bab ketiga disajikan pembahasan mengenai sistem yang dibangun. Pada sub-bab kelima disajikan hasil penelitian dan pembahasan analisa dari hasil penelitian. Dan pada sub-bab kelima menyajikan kesimpulan dari penelitian yang telah dilakukan.

2. Studi Terkait

2.1 Sistem Pemberi Rekomendasi

Sistem pemberi rekomendasi adalah sebuah sistem berupa perangkat lunak yang dibangun untuk menghasilkan keluaran berupa sejumlah *item* teratas berdasarkan nilai yang telah diperhitungkan menggunakan sejumlah algoritma. Adapun sejumlah *item* yang dihasilkan tersebut merupakan keluaran atas permintaan pengguna yang membutuhkan dan sebelumnya pengguna telah memberikan kriteria tertentu dari sejumlah *item* yang diminta. Sistem pemberi rekomendasi yang digunakan dalam penelitian itu bersifat *personalized*, yaitu pengguna yang berbeda akan mendapatkan saran rekomendasi *item* yang beragam dan ditampilkan dalam bentuk urutan peringkat *item* [10].

Terdapat 3 jenis objek data yang digunakan dalam sistem pemberi rekomendasi, yaitu *users*, *items*, dan *transactions*. *Users* adalah pengguna sistem pemberi rekomendasi. Berbagai informasi mengenai *user* perlu diekploitasi oleh sistem pemberi rekomendasi agar bisa mempersonalisasi rekomendasi dan interaksi manusia-komputer. Sedangkan *items* adalah objek yang direkomendasikan. Karakter *item* mungkin bisa dilihat dari kompleksitas dan nilai kegunaannya. Misalnya jika *item* bernilai positif maka *item* berguna bagi *user* atau sebaliknya jika *item* bernilai negatif maka *item* tidak cocok karena *user* mungkin membuat kesalahan dalam menentukan pilihannya. Sedangkan *transactions* adalah data yang digunakan oleh sistem dalam algoritma pembuatan rekomendasi dan berupa seperti *log* penyimpanan informasi penting yang dihasilkan dari interaksi manusia-komputer. Data *transactions* paling populer yang dikumpulkan sistem pemberi rekomendasi adalah data *rating item* yang diberikan oleh *user* dan dapat dikumpulkan secara *explicit* atau *implicit* [10].

Dalam [9], dijelaskan bagaimana pentingnya data umpan balik dari pengguna seperti data yang disukai dan tidak disukai pengguna digunakan untuk membangun sistem pemberi rekomendasi. Karena data tersebut sangat membantu untuk sistem pemberi rekomendasi dapat memberikan saran *item* yang relevan kepada pengguna. Terdapat 2 mekanisme umpan balik pengguna yang dapat dijadikan data sebagai ketersediaan bahan yang diperlukan dalam membangunnya, yaitu *explicit feedback* dan *implicit feedback*. *Explicit feedback* merupakan data umpan balik yang diberikan oleh pengguna secara sadar (*explicit*) terhadap suatu *item*. Jadi dalam hal ini, pengguna secara sadar memberikan datanya ke dalam sistem, contohnya seperti pengguna memberikan *rating* berbentuk bintang pada suatu *item*, menekan tombol suka untuk suatu *item*, atau memberikan komentar dan ulasan pendapatnya mengenai *item* tersebut. Sedangkan *implicit feedback* merupakan data umpan balik yang diberikan oleh pengguna secara tidak sadar (*implicit*). Aktivitas yang dilakukan oleh pengguna dalam berselancar di suatu situs web adalah salah satu contoh *implicit feedback*. Hal itu dikarenakan pengguna tidak menyadari bahwa aktivitas yang dilakukannya tersebut adalah data umpan balik yang akan tersimpan di sistem dan bisa diambil kapanpun saat data tersebut memang diperlukan. Dalam penelitian yang saya lakukan ini, data umpan balik pengguna yang saya gunakan berbentuk *explicit feedback* yang berupa kumpulan data *rating* yang diberikan oleh pengguna terhadap *item* anime.

Sistem pemberi rekomendasi memiliki 6 kelas pendekatan rekomendasi yang berbeda [10]: *collaborative filtering*, *content-based*, *demographic*, *knowledge-based*, *community-based*, dan *hybrid recommender system*. Pendekatan *collaborative filtering* membuat rekomendasi *item* yang disukai oleh *user* lain di riwayatnya kepada *user* yang membutuhkan rekomendasi *item* dan memiliki selera yang sama dengannya. Dalam pendekatan *collaborative filtering*, terdapat 2 metode yang dapat digunakan, yaitu metode *user-based collaborative filtering* dan *item-based collaborative filtering*. Metode *user-based* utamanya mengandalkan pendapat dari *user* yang memiliki pikiran yang sama terkait *item* untuk membuat prediksi *rating item*. Sedangkan metode *item-based* utamanya mengandalkan *rating* dari *item* yang memiliki kemiripan. Untuk pendekatan *content-based*, sistem membuat rekomendasi *item* yang memiliki banyak kemiripan dengan *item* yang pernah disukai oleh *user* di riwayatnya. Berbeda dengan kedua pendekatan tersebut, pendekatan *demographic* membuat rekomendasi *item* yang berbeda untuk kelompok demografi yang berbeda. Sementara itu, pendekatan *knowledge-based* membuat rekomendasi *item* dengan pengetahuan domain tertentu sehingga berguna untuk *user* dan dapat memenuhi kebutuhan preferensinya. Lalu untuk pendekatan *community-based*, sistem membuat rekomendasi *item* dengan melihat preferensi dari temannya *user* alih-alih preferensi dari orang yang tidak dikenal sama sekali oleh *user*. Dan pendekatan *hybrid recommender system* membuat rekomendasi dengan mengkombinasikan 2 atau lebih pendekatan rekomendasi lain dengan tujuan dapat memanfaatkan keunggulan dari salah satu pendekatan rekomendasi agar dapat menutupi kelemahan rekomendasi lainnya. *Hybrid recommender system* adalah pendekatan yang saya gunakan dalam melakukan penelitian. Adapun dalam hal ini, saya mencoba mengkombinasikan pendekatan *content-based* dengan pendekatan *item-based collaborative filtering*.

2.2 *K-Means Clustering*

Clustering yang juga disebut sebagai *unsupervised learning*, merupakan metode dalam menetapkan *item* ke dalam beberapa kelompok sehingga *item* dalam kelompok yang sama memiliki kemiripan yang lebih tinggi daripada *item* dalam kelompok yang berbeda [6] [10]. Algoritma *k-means clustering* merupakan salah satu algoritma *clustering* yang dianggap paling luas. Algoritma ini bertujuan dapat mengelompokkan data yang memiliki kemiripan ke dalam *cluster* yang sama. Agar dapat mengimplementasikan algoritma *k-means*, dalam dataset ada suatu hal yang disebut sebagai *centroid*. *Centroid* adalah elemen kunci dari algoritma. *K-means* bergantung pada *centroid* untuk mendeteksi data yang paling dekat dengan masing-masing *centroid*, dan perulangan ini berlanjut hingga tidak ada peningkatan lebih lanjut dalam jarak antara data serupa, yakni hingga algoritma konvergen [12]. Algoritma (1) yang merangkum langkah dalam algoritma *k-means clustering*:

Algorithm 1 Konsep Algoritma *K-Means*

1. Tentukan jumlah kelompok (K) yang ingin dibentuk.
2. Pilih secara acak K titik awal sebagai *centroid*.
3. Hitung jarak antara setiap data dengan masing-masing *centroid*.
4. Masukkan setiap data ke dalam kelompok dengan *centroid* terdekat.
5. Perbarui posisi *centroid* dengan menghitung rata-rata posisi data dalam setiap kelompok.
6. Ulangi langkah 3-5 hingga tidak ada perubahan dalam posisi *centroid* atau konvergen.

Penentuan jumlah *optimal k cluster* yang akan digunakan dalam mengidentifikasi adalah hal yang penting untuk dilakukan agar dapat menerapkan algoritma *k-means* dengan benar. *Elbow method* merupakan salah satu metode paling berhasil untuk digunakan Ketika ingin menghitung jumlah *k optimal cluster* [12]. *Elbow method*

adalah metode yang melihat fungsi berupa persentase varian dari jumlah *cluster*. Gagasan dari metode ini yaitu, seseorang harus memilih sejumlah *cluster* sehingga menambahkan *cluster* lain tidak memberikan pemodelan data yang jauh lebih baik. Persentase varian yang dijelaskan oleh *cluster* diplot terhadap jumlah *cluster*. *Cluster* pertama akan menambahkan banyak informasi tetapi pada titik tertentu keuntungan marjinal akan turun drastis dan memberikan *angle* pada grafik. 'K' yang tepat yaitu jumlah *cluster* dipilih pada titik tersebut, oleh karena itu disebut *"Elbow criterion"* [2].

2.3 TF-IDF

Sistem *content-based* pada dasarnya dikembangkan untuk menyaring dan merekomendasikan *item* berbasis teks seperti pesan *email* atau berita. Sehingga mempertahankan *list* fitur "meta-informasi" bukan pendekatan standar dalam sistem *content-based*, melainkan menggunakan *list keyword* relevan yang muncul dalam dokumen yang dihasilkan secara otomatis dari konten dokumen itu sendiri atau dari deksripsi teks bebasnya [8].

Salah satu cara mengkodekan dokumen dalam *list keyword* adalah membuat *list* semua kata yang muncul dalam semua dokumen dan mendeskripsikan setiap dokumen dengan *vektor boolean*. Namun pendekatan *boolean* sederhana ini memiliki kekurangan masalah yang bisa menyebabkan sistem pemberi rekomendasi akan menjadi cenderung mengajukan dokumen yang panjang. Hal ini bisa diatasi dengan mendeskripsikan dokumen menggunakan format pengkodean TF-IDF [8].

TF-IDF merupakan akronim gabungan dari TF dan IDF. TF (*Term-Frequence*) adalah banyaknya jumlah token muncul dalam dokumen korpus yang dibagi dengan jumlah total token [9]. TF diilustrasikan dalam rumus persamaan (1) berikut:

$$tf_{i,j} = \frac{\text{Jumlah kemunculan } term\ i \text{ dalam dokumen j}}{\text{Jumlah total term dalam dokumen j}} \tag{1}$$

Sedangkan IDF (*Inverse Document Frequency*) adalah *log* banyaknya jumlah total dokumen korpus dalam keseluruhan dokumen yang telah dibagi dengan jumlah keseluruhan dokumen dengan kata terpilih [9]. IDF diilustrasikan dalam rumus persamaan (2) berikut:

$$idf_i = log(\frac{\text{Jumlah total dokumen}}{\text{Jumlah dokumen yang mengandung term i}}) \tag{2}$$

Sehinggga TF dan IDF jika kedua hal ini dikalikan bisa digunakan dalam menghitung kerelevanan sebuah kata yang terkandung dalam dokumen. Tingkat karakterisasasi dalam dokumen bisa ditunjukkan melalui nilai TF-IDF dari suatu kata. Kata yang terkandung dalam sebuah dokumen akan menjadi bernilai penting jika memiliki nilai TF-IDF yang tinggi [4]. TF-IDF diilustrasikan dalam rumus persamaan (3) berikut:

$$tfidf_{i,j} = tf_{i,j} \cdot idf_i \tag{3}$$

2.4 *Cosine Similarity*

Untuk mengetahui kedekatan antara dua dokumen, maka dibutuhkan *similarity measure*. *Cosine similarity* adalah salah satu contoh *similarity measure* yang paling umum digunakan. *Similarity measure* ini juga umum digunakan dalam bidang pencarian informasi dan *text mining* untuk membandingkan 2 dokumen teks, Dimana dokumen direpresentasikan sebagai *vector of terms* [8] [10].

Cosine similarity mengukur kemiripan antara 2 vektor berdimensi n berdasarkan sudut di antara keduanya. Ini merupakan hasil perkalian *dot product* 2 vektor yang kemudian dibagi dengan perkalian panjang (besaran) 2 vektor [8] [9]. Untuk lebih jelasnya *cosine similarity* digambarkan dalam persamaan (4) berikut:

$$sim(A,B) = cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} \tag{4}$$

Keterangan:

- A = Vektor *item* A
- B = Vektor *item* B
- θ = sudut antara vektor *item* A dan vektor *item* B
- $\|A\|$ = Panjang vektor A
- $\|B\|$ = Panjang vektor B

2.5 *Item-Based Collaborative Filtering*

Dalam pendekatan *collaborative filtering*, terdapat 2 pendekatan utama untuk menghubungkan antara *item* dengan *user*, yaitu pendekatan *neighborhood* dan *model latent factor*. Contoh pendekatan *neighborhood* salah satunya adalah *item-based filtering*. Pendekatan *item-based* membuat model berdasarkan preferensi *user* terhadap *item* dengan melihat *rating item* serupa yang diberikan oleh *user* yang sama. Dalam sistem rekomendasi, ketika jumlah *user* ternyata melebihi dari jumlah *item* yang tersedia, pendekatan ini bisa jadi dapat memberikan rekomendasi yang lebih akurat dan efisien secara komputasi dan membutuhkan sedikit pembaruan. Sedangkan pendekatan model latent factor, seperti *matrix factorization* (contoh: SVD) membuat pendekatan alternatif dengan memetakan *item* dan pengguna ke dalam ruang faktor laten yang sama yang menjelaskan rating dengan mengkarakterisasi *item* dan *user* berdasarkan faktor-faktor yang diidentifikasi dari *user feedback* [10].

Meskipun kedua pendekatan tersebut berkembang dari prinsip dasar yang berbeda, namun keduanya masih memiliki banyak kesamaan karena keduanya adalah model linear [10]. Sehingga dalam penelitian ini, peneliti memilih menggabungkan pendekatan item-based filtering dengan matrix factorization sebagai pendekatan untuk mereduksi dimensi (seperti algoritma SVD) sebagai salah satu pendekatan *filtering* dalam sistem *hybrid filtering* anime yang dibangun. Tujuan pemilihan pendekatan ini adalah untuk memanfaatkan penggabungan kekuatan dari kedua model *linear* tersebut.

2.6 Algoritma SVD

Dalam buku [10] dijelaskan bahwa, *Singular Value Decomposition* (SVD) merupakan pendekatan matematis dan salah satu pendekatan Matrix Factorization (MF) yang sering digunakan untuk mereduksi dimensi dalam berbagai konteks. Konsep dari SVD memecah matriks data menjadi 3 komponen matriks. Dekomposisi ini memungkinkan pemetaan data ke dalam ruang dimensi yang lebih rendah mewakili "*concept-concept*" yang bisa dihitung kekuatannya dalam konteks dataset. Ketiga komponen matriks tersebut diantaranya: (1) matriks U (*item-to-concept similarity*), (2) matriks λ (*strength of each concept*), dan (3) matriks V^T (*term-to-concept*).

Algoritma SVD bermanfaat juga dalam pendekatan collaborative filtering, terutama dalam mengungkap hubungan laten antara *user* dan item. Dalam konteks ini, terdapat 2 tujuan penggunaan SVD. Tujuan pertama adalah untuk menemukan hubungan laten dengan memfaktorkan matriks *user-item* setelah melalui beberapa tahap normalisasi. Dan tujuan kedua adalah memanfaatkan ruang berdimensi rendah yang dihasilkan dari SVD untuk memperbaiki pembentukan ketetanggan (*neighborhood*) dalam pendekatan KNN [10]. Tujuan pertama merupakan tujuan yang sama dengan tujuan penulis dalam memanfaatkan algoritma SVD untuk digunakan sebagai *matrix factorization* untuk mereduksi dimensi pada pembangunan model *item-based filtering*.

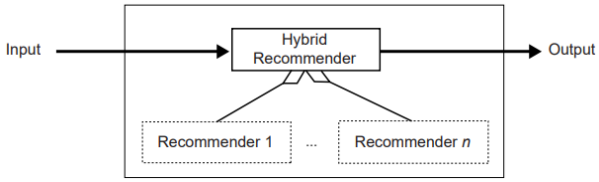
2.7 Hybrid Filtering

Di dalam penelitian ini, konsep untuk sistem pemberi rekomendasi yang dibangun adalah menggunakan pendekatan hybrid filtering. *Hybrid filtering* merupakan pendekatan yang mengkombinasikan antara 2 atau lebih dari macam-macam teknik rekomendasi. Pendekatan ini bertujuan untuk bisa mendapatkan kinerja yang lebih baik dengan meminimalkan kekurangan dari beberapa pendekatan filtering yang berdiri sendiri [3]. Contoh masalah umum dalam sistem pemberi rekomendasi, seperti cold start, sparsity dan scalability. Ketiga masalah tersebut biasa ditemukan dalam pendekatan collaborative filtering. Cold start adalah kondisi masalah ketika terdapat *user* baru yang belum pernah memberikan rating pada item. Karena tidak adanya rating dari *user* baru menyebabkan sistem bisa kesulitan menemukan informasi preferensi dari *user* baru tersebut. Sparsity adalah kondisi masalah kelangkaan (sparse). Kelangkaan data yang dimaksudkan disini yaitu matriks data berisi data yang tidak lengkap dan hal ini dapat menghasilkan nilai similarity antar *item* kecil sehingga bisa menyebabkan kualitas rekomendasi yang buruk. Sedangkan scalability adalah kondisi masalah data yang digunakan berskala besar sehinnga membuat sistem memerlukan peningkatan daya komputasi besar untuk dapat memberikan rekomendasi dengan waktu yang digunakan lebih efisien [5].

Dijelaskan dalam buku [8], terdapat 7 varian pendekatan *hybrid filtering* yang terbagi ke dalam 3 basis design, diantaranya:

1. *Monolithic Hybridization Design*

   Desain hybrid monolithic ini memiliki konsep dimana beberapa pendekatan sistem pemberi rekomendasi berkonrtribusi secara virtual karena penggabungan yang menggunakan data input tambahan yang spesifik untuk algoritma rekomendasi lain. Atau data input tersebut bisa juga ditambah dengan 1 teknik dan secara faktual dieksploitasi oleh teknik lain. Gambar (1) menunjukkan bagaimana alur desain *hybrid monolithic* ini dibuat. Ada 2 pendekatan *hybrid* yang menggunakan desain *hybrid monolithic* ini, diantaranya:



Gambar 1. *Monolithic Hybridization Design*

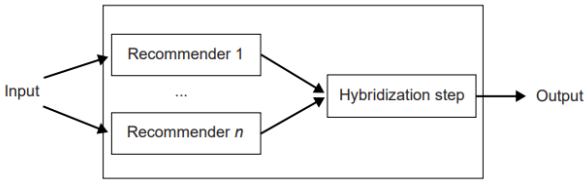
(a) *Feature Combination Hybrid*

Pendekatan ini merupakan komponen rekomendasi desain *monolithic* yang memerlukan beragam *data input*. Dan sesuai dengan namanya, pendekatan ini memanfaatkan fitur-fitur dari pendekatan sistem pemberi rekomendasi untuk bisa saling dikombinasikan dan menghasilkan rekomendasi *item*.

(b) *Feature Augmentation Hybrid*

Pendekatan ini merupakan komponen rekomendasi desain *monolithic* yang dapat digunakan untuk mengintegrasikan beberapa algoritma rekomendasi. Berbeda dengan *feature combination*, pendekatan *hybrid* ini tidak hanya menggabungkan dan memproses beberapa jenis *input* saja, namun juga menerapkan beberapa langkah transformasi yang lebih kompleks.

2. *Parallelized Hybridization Design*



Gambar 2. *Parallelized Hybridization Design*

Konsep desain *hybrid paralel* adalah memparalelkan sistem pemberi rekomendasi dapat beroperasi secara independen satu sama lain dan menghasilkan daftar rekomendasi *item* yang terpisah. Proses desain *hybrid* ini terletak pada penggabungan *output* dari beberapa sistem pemberi rekomendasi menjadi serangkaian rekomendasi *item* akhir. Gambar (2) menunjukkan bagaimana alur desain *hybrid monolithic* ini dibangun. Ada 3 pendekatan *hybrid* yang menggunakan desain *hybrid paralel* ini, diantaranya:

(a) *Mixed Hybrid*

Pendekatan *mixed hybrid* merupakan strategi *hybridisasi* yang menggabungkan hasil dari beberapa sistem pemberi rekomendasi yang berbeda pada tingkat *user interface*. Hasil dari beberapa teknik rekomendasi yang berbeda akan disajikan bersama-sama.

(b) *Weighted Hybrid*

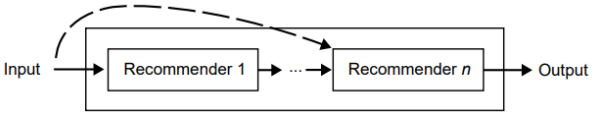
Pendekatan *weighted hybrid* ini juga menggabungkan rekomendasi dari 2 atau lebih teknik rekomendasi. Namun berbeda dengan *mixed hybrid*, pendekatan ini menggabungkan teknik rekomendasi bukan pada hasil rekomendasi masing-masing teknik, namun dengan menghitung jumlah skor tertimbang (*weighted sum score*).

(c) *Switching Hybrid*

Pendekatan *switching hybrid* ini berbeda dengan kedua pendekatan *hybrid* dengan desain *paralel* lainnya. Pendekatan ini memerlukan oracle yang memutuskan sistem pemberi rekomendasi mana yang harus digunakan dalam situasi tertentu. Jadi pemilihan teknik rekomendasi dan hasilnya akan bergantung pada profil *user* dan/atau kualitas hasil rekomendasi.

3. *Pipelined Hybridization Design*

Berbeda dengan kedua desain *hybrid* lainnya, desain ini memiliki arstitektur yang terlihat seperti membentuk saluran pipa dari sistem pemberi rekomendasi yang digabungkan secara bersama yang bisa dilihat seperti pada gambar (3). Dalam desain *pipelined* ini, *output* dari satu pendekatan sistem pemberi rekomendasi dapat menjadi bagian dari *input* untuk pendekatan sistem pemberi rekomendasi berikutnya. Ada 2 pendekatan *hybrid* yang menggunakan desain *hybrid pipelined* ini, diantaranya:



Gambar 3. *Pipelined Hybridization Design*

(a) *Cascade Hybrid*

Pendekatan *cascade hybrid* didasari pada urutan teknik rekomendasi yang berurutan. Dimana setiap teknik rekomendasi berikutnya hanya menyempurnakan rekomendasi pendahulunya. Hasil rekomendasi teknik penerus juga dibatasi pada *item* yang juga direkomendasikan oleh teknik sebelumnya.

(b) *Meta-level Hybrid*

Berbeda dengan pendekatan *cascade hybrid*, dalam pendekatan *meta-level* ini salah satu dari sistem pemberi rekomendasi akan membuat model sistem. Lalu dari model sistem tersebut akan dimanfaatkan oleh sistem pemberi rekomendasi utama untuk membuat hasil rekomendasi.

3. Sistem yang Dibangun

3.1 Diagram Alur Sistem

Uji coba penelitian terbagi ke dalam 3 skenario yang berbeda. Adapun skenario pertama adalah melakukan uji coba *hybrid filtering* dengan menggunakan 3 kombinasi *filtering* yang terurut diantaranya ada *filtering k-means clustering* menggunakan *elbow method*, *filtering* TF-IDF dengan *cosine similarity*, dan terakhir *item-based filtering* menggunakan algoritma SVD. Rancangan diagram alur sistem untuk skenario pertama dapat dilihat pada gambar (4).

Berbeda dengan skenario pertama, skenario kedua melakukan uji coba *hybrid filtering* dengan hanya menggunakan 2 kombinasi *filtering* yang terurut diantaranya ada *filtering k-Means clustering* menggunakan *elbow method*, dan terakhir *item-based filtering* menggunakan algoritma SVD. Rancangan diagram alur sistem untuk skenario kedua dapat dilihat pada gambar (5).

Dan skenario ketiga sama dengan skenario kedua yang melakukan uji coba *hybrid filtering* dengan hanya menggunakan 2 kombinasi *filtering*. Namun 2 *filtering* terurut yang diterapkan diantaranya ada *filtering* TF-IDF dengan *cosine similarity* dan *item-based filtering* menggunakan algoritma SVD. Rancangan diagram alur sistem untuk skenario ketiga dapat dilihat pada gambar (6)

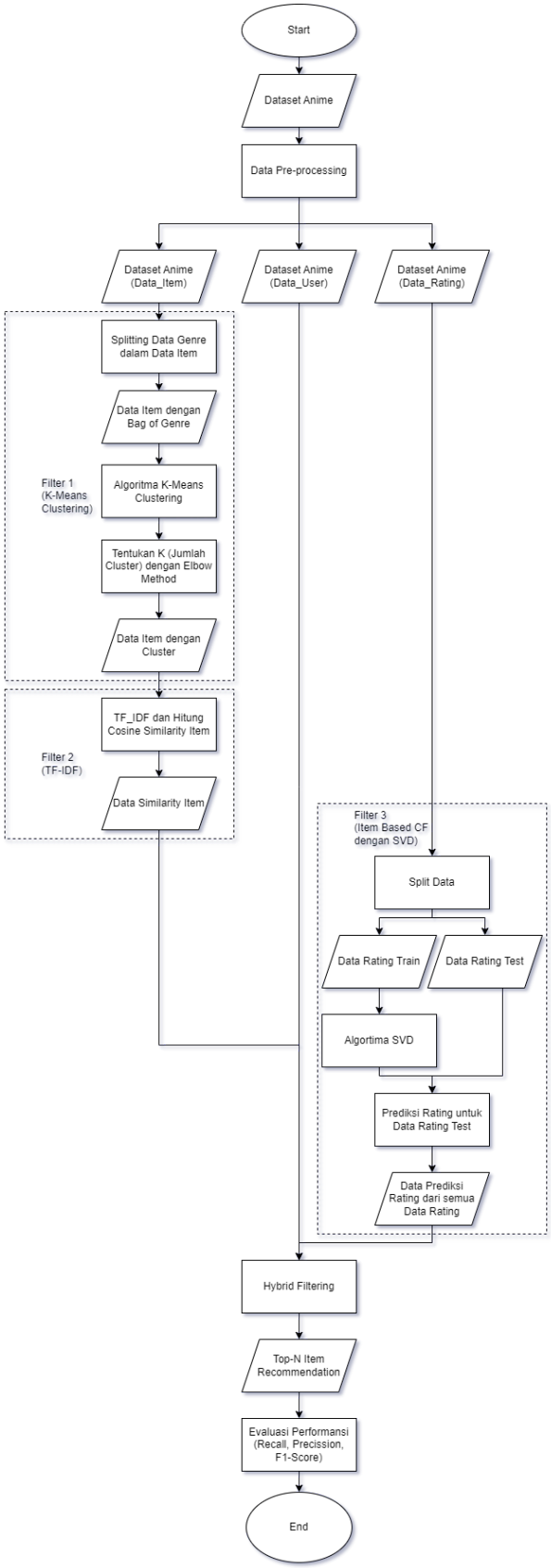
Pada semua skenario sistem ini memiliki beberapa tahapan yang dimulai dari pengelolaan dataset dalam tahap *data pre-processing*. Lalu pada tahap selanjutnya adalah melakukan *filtering item* secara terurut dimana masing-masing skenario menerapkan pemilihan *filtering* terurut yang berbeda-beda. Memasuki tahap berikutnya ialah *hybrid filtering* untuk mendapatkan *top-N item recommendation*. Dan pada tahap terakhir, dilakukan evaluasi performansi sistem *hybrid filtering* yang dibangun menggunakan beberapa metrik pengukuran evaluasi kinerja model sistem yang diantaranya adalah *recall*, *precission*, dan *f1-score*.

3.2 Dataset

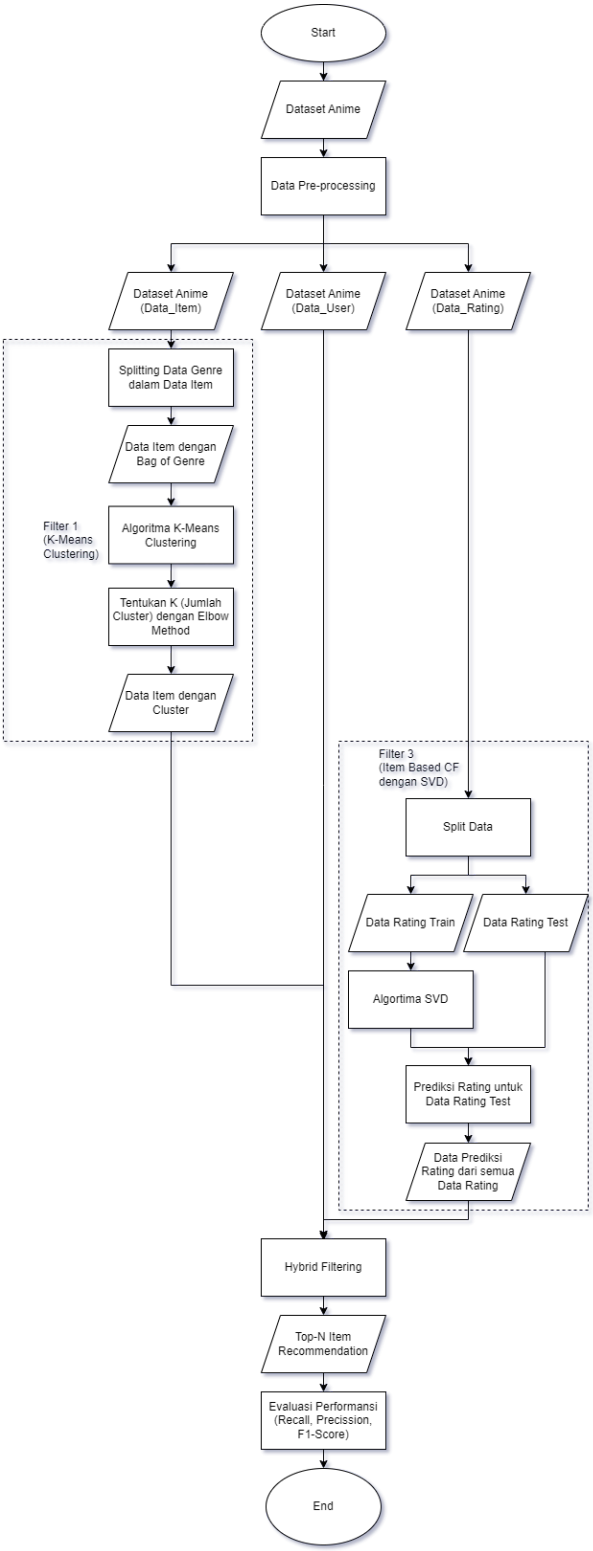
Dalam penelitian ini digunakan dataset Anime yang diunduh dari situs *web* Kaggle. Dataset Anime memiliki informasi diantaranya ada data *item* anime yang berjumlah 16216 *item*, data profil *user* yang berjumlah 47885 *user*, dan data ulasan *rating user* yang berjumlah 130519 ulasan. *Dataset* anime tersebut sebelumnya merupakan *dataset* yang telah dilakukan proses *crawling* dari situs *web* myanimelist.net dan dibagikan secara open source melalui situs *web* Kaggle [1].

Dataset anime ini terbagi ke dalam 3 *file* yang berbeda. *File-file* tersebut beserta informasinya sebagai berikut:

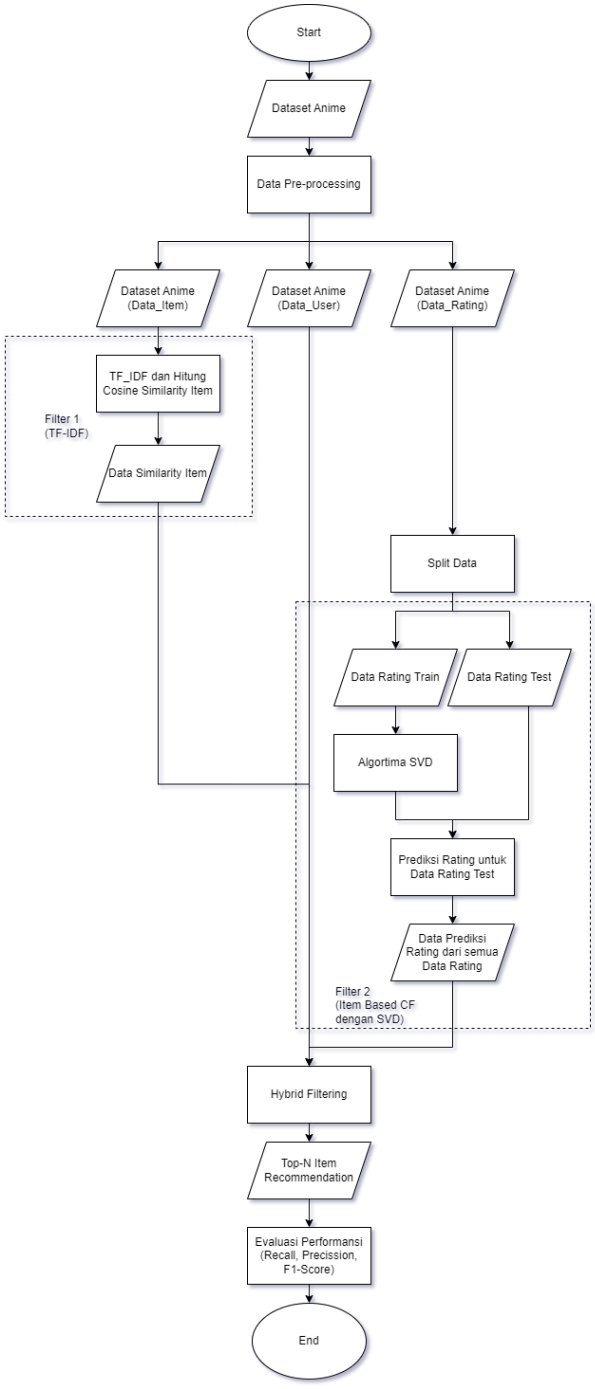
1. animes.csv: *File* ini berisikan daftar informasi tentang *item* anime. Adapun *file* ini berupa tabel data dengan kolom data diantaranya adalah
    - uid (*Id item* anime),
    - title (Judul anime),
    - synopsis (sinopsis anime),
    - genre (*genre* anime),



Gambar 4. Diagram Alur Sistem *Hybrid Filtering* yang Dibangun - Skenario 1



Gambar 5. Diagram Alur Sistem *Hybrid Filtering* yang Dibangun - Skenario 2



Gambar 6. Diagram Alur Sistem *Hybrid Filtering* yang Dibangun - Skenario 3

- aired (Tanggal tayang anime),
- episodes (Jumlah episode anime),
- members (tidak diketahui informasi mengenai ini),
- popularity (tidak diketahui informasi mengenai ini),
- ranked (urutan peringkat *item*),
- score (rating rata-rata *item*),
- `img_url` (*link* gambar *item*),
- link (*link url item*).

2. profiles.csv: *File* ini berisikan informasi tentang *user* yang menonton anime. Adapun *file* ini berupa tabel data dengan kolom data diantaranya adalah

   - profile (*Id* profil unik *user*),
   - gender (jenis kelamin *user*),
   - birthday (tanggal lahir *user*),
   - `favorites_anime` (referensi id anime yang disukai *user*),
   - link (*link url user*).

3. reviews.csv: *File* ini berisikan informasi mengenai ulasan yang diberikan oleh *user* terhadap anime berupa bentuk *rating*. Adapun *file* ini berupa tabel data dengan kolom data diantaranya adalah

   - uid (*Id* ulasan),
   - profile (*Id* profil unik *user*),
   - `anime_uid` (Id *item* anime),
   - text (teks ulasan),
   - score (skor *rating* keseluruhan),
   - scores (semua rincian skor *rating*),
   - link (*link url* ulasan).

Namun dari keseluruhan jumlah dataset yang tersedia, dalam penelitian ini hanya akan menggunakan 20% dari data *item* anime, dan 20% dari data *user* yang dipilih secara acak, serta untuk data ulasan yang digunakan menyesuaikan dari id *item* anime dan id profil unik *user* yang terpilih dalam data *item* anime dan data *user* setelah pengurangan jumlah *dataset*. Pengurangan jumlah *dataset* yang digunakan untuk penelitian ini menyisakan data *item* anime berjumlah 3243 item, data profil *user* berjumlah 9577 *user*, dan data ulasan *rating user* berjumlah 5428 ulasan. Alasan utama dari pengurangan *dataset* yang digunakan untuk penelitian ini adalah untuk meningkatkan efisiensi *runtime program* dan untuk mengatasi masalah *scalability*. Dengan menggunakan *dataset* yang lebih kecil, waktu yang dibutuhkan untuk menjalankan program dapat dikurangi secara signifikan, sehingga tidak memerlukan waktu yang lama dalam proses eksekusi. Langkah ini juga memungkinkan untuk melakukan lebih banyak eksperimen dan iterasi dalam waktu yang lebih singkat, yang pada gilirannya dapat mempercepat pengembangan dan pengujian algoritma yang digunakan.

3.3 Eksperimen

3.3.1 *Data Pre-processing*

Sebelum memulai penelitian, ada baiknya dilakukan *data pre-processing* agar data yang digunakan bisa membuat program yang dibangun menjadi lebih efisien dan sesuai kebutuhan penelitian. Pada tahap *data pre-processing* ada beberapa hal yang dilakukan. Pertama dilakukan penghapusan kolom data yang tidak diperlukan untuk penelitian. Kemudian dilakukan penamaan ulang nama kolom data agar lebih mudah dibaca dan dipahami, Selanjutnya merapihkan beberapa isi kolom data agar mudah diolah dalam program, contohnya seperti pada Tabel (1). Dilakukan juga penghapusan baris data duplikat karena bisa membuat program tidak efisien. Serta pengurangan jumlah *dataset* juga dilakukan di dalam tahap *data pre-processing* ini.

Tabel 1. **Contoh perapihan isi kolom data *'Genre'***

Sebelum Data Pre-processing	Setelah Data Pre-Processing
['Comedy', 'Sports', Drama', 'School', 'Shounen']	Comedy, Sports, Drama, School, Shounen
['Drama', 'Music', 'Romance', 'School', 'Shounen']	Drama, Music, Romance, School, Shounen
['Sci-Fi', 'Adventure', 'Mystery', 'Drama', 'Fantasy']	Sci-Fi, Adventure, Mystery, Drama, Fantasy

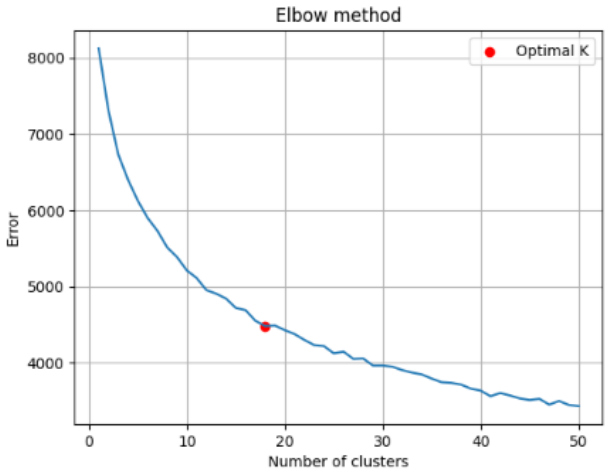
3.3.2 *K-Means Clustering*

Memasuki tahap pertama penelitian ini, dilakukan pengelompokan data *item* berdasarkan kelompok *cluster* tertentu menggunakan algoritma *k-means clustering*. Sebelum itu, dari data *item* akan dilakukan *split data* untuk menyimpan kolom *data genre* untuk dibuatkan *bag of genre* dari *item*, yang contohnya dapat dilihat seperti pada tabel (2). *Bag of genre* ini berisikan data *item* dengan *genre* yang disesuaikan dengan nilai *biner*. Apabila *item* memiliki *genre* yang sesuai maka diisi nilai 1, jika sebaliknya maka diisi nilai 0.

Tabel 2. **Contoh *bag of genre* untuk data *item* anime**

Anime_ID	Genre	Adventure	Fantasy	Kids	⋯	Music	Romance
39694	Music	0	0	0	⋯	1	0
37194	Music, Kids	0	0	1	⋯	1	0
11583	Music	0	0	0	⋯	1	0
6094	Fantasy, Kids	0	1	1	⋯	0	0

Setelah membuat *bag of genre*, disini *bag of genre* dipakai sebagai fitur untuk mengelompokkan data berdasarkan *cluster* menggunakan algoritma *k-means clustering*. Namun sebelum itu, ditentukan terlebih dahulu jumlah *cluster k* yang ingin diterapkan. Agar jumlah *cluster k* dapat membuat model sistem menjadi lebih baik, cara menentukan *optimal k* nya adalah dengan menggunakan *elbow method*. Dan saat *elbow method* dijalankan, jumlah *optimal k* yang didapat adalah berada di angka 18 seperti yang ditunjukkan pada gambar (7). Sehingga pada algoritma *k-means clustering*, dari data *item* yang ada, masing-masing *item*-nya akan dikelompokkan ke dalam 18 *cluster* yang berbeda-beda dan datanya disimpan ke dalam data *item* baru yang sudah dilabeli nomor *cluster*.



Gambar 7. *Elbow Method* - Menentukan *Optimal K Cluster*

3.3.3 TF-IDF dan *Cosine Similarity*

Dalam tahap ini, langkah pertama yang perlu dilakukan adalah melakukan *data cleaning* pada fitur *data synopsys item* anime. Proses *data cleaning* digunakan untuk mengubah format data menjadi format yang lebih baik dari ketidaklengkapan dan seberapa tidak beraturannya data. *Data cleaning* yang dilakukan diantaranya adalah penghapusan data dari *stopwords* dan simbol menggunakan *library* nltk. Tabel (3) adalah contoh bagaimana *input* dan *output* dari proses *data cleaning* ini.

Tabel 3. **Contoh *data cleaning* pada fitur 'Anime_Synopsis' dari data *item* anime**

Anime_Id	*Input*	*Output*
39694	A collaborative web commercial between the Japanese sports drink brand Pocari Sweat and the 2017 Fuji Rock Festival. It was posted to their official Facebook page May of 2017.	Collaborative web commercial Japanese sports drink brand Pocari Sweat 2017 Fuji Rock Festival posted official Facebook page May 2017
37194	A Minna no Uta music video sung by Keiko Utsumi and animated by Nanke Kouji. It has no relation to the Minna no Uta music video from 2010 which shares the name name.	Minna Uta music video sung Keiko Utsumi animated Nanke Kouji relation Minna Uta music video 2010 shares name name

Setelah dilakukan *data cleaning*, data siap untuk masuk ke tahap utama TF-IDF, yaitu mengubah teks yang telah diproses menjadi representasi numerik. Adapun proses ini menggunakan TF-IDF *vectorizer* untuk menghitung TF-IDF dari teks, dan TF-IDF *Matrix* untuk membuat matriks dari representasi numerik dari setiap dokumen. Setelah matriks didapatkan, matriks dinormalisasi untuk kemudian melakukan proses perhitungan kemiripan antar *item* menggunakan *cosine similarity*.

3.3.4 *Item-Based Collaborative Filtering* Menggunakan Algoritma SVD

Pada tahap ini, dilakukan *filtering* berbasis *item* (*item-based filtering*) dengan cara mencari kesamaan antar *item* yang mungkin relevan untuk memberikan rekomendasi item. Sebelum itu, data ulasan rating dibagi menjadi dua bagian: 80% digunakan sebagai *data train* dan sisanya 20% sebagai *data test*. Pembagian data ini bertujuan untuk membuat prediksi *rating* dari model yang dibangun menjadi lebih akurat melalui proses pelatihan dan pengujian model.

Tabel 4. **Contoh Prediksi *Rating Item***

User_Id	Anime_Id	Rating_by_User	Predicted_Rating
Sonicfanx1	31845	7	6.828807
NJ_animeluvr	28121	8	7.041153
grazr	21507	5	5.72924
mow	32668	8	7.151256
molesys	2981	8	7.283954

Untuk melatih model sistem, *data train* digunakan dalam proses *matrix factorization* dengan algoritma SVD. Setelah model dilatih, model yang telah dibangun dilakukan pengujian menggunakan data test untuk memprediksi rating item. Tabel (4) merupakan contoh hasil prediksi *rating item* yang didapat dari model SVD yang telah diuji menggunakan *data test*.

3.3.5 Hybrid Filtering

Tahapan utama dalam penelitian ini adalah proses hibridisasi filtering rekomendasi item. Semua pendekatan filtering yang digunakan oleh peneliti dalam melakukan filtering *item* anime dikombinasikan menjadi satu sistem yang disebut *hybrid filtering*. Namun karena uji coba penelitian terbagi ke dalam 3 skenario, penerapan *hybrid filtering* pada masing-masing skenario akan berbeda-beda dalam penggunaan jumlah *filtering* serta penerapan pendekatan filtering apa yang digunakan. Adapun sistem *hybrid filtering* dalam penelitian ini menerapkan pendekatan *cascade hybrid* yang menggunakan *desain pipelined hybridization*.

Pertama, untuk dapat bisa menjalankan program, simpan query_user_id dan query_item_id yang diambil secara acak untuk mendapatkan rekomendasi *item* anime untuk *user* dengan query_user_id tersebut yang memiliki kemiripan *item* dengan query_item_id. Selanjutnya memasuki tahapan *hybrid filtering*, pada skenario pertama sistem *hybrid filtering* akan melakukan *filtering item* yang pertama menggunakan *k-means clustering*. Dalam *filtering k-means clustering* ini, sistem akan mengambil semua *item* yang berada di *cluster* yang sama dengan *item* query_item_id. *item* yang telah didapatkan dari hasil *filtering* pertama disimpan untuk dilakukan *filtering* kedua. Pada *filtering* {*item* yang kedua, peneliti memilih menggunakan TF-IDF sebagai alat *filtering* untuk kemudian dicari kesamaan *item* menggunakan *cosine similarity*. Dari *filtering item* yang kedua ini, diambil *top*-100 *item* anime dengan kemiripan antara *item* dengan query_item_id tertinggi dan disimpan sebelum dilakukan *filtering* yang terakhir. Dan memasuki *filtering* yang terakhir, peneliti memilih menggunakan pendekatan *item-based filtering*

dengan algoritma *Singular Value Decomposition* (SVD) sebagai metode *matrix factorization* untuk memprediksi *rating item*.

Berbeda tahapan *hybrid filtering* pada skenario kedua, *hybrid filtering* akan melakukan *filtering item* yang pertama menggunakan k-means clustering. Dalam filtering k-means clustering ini, sistem akan mengambil semua *item* yang berada di *cluster* yang sama dengan *item* query_item_id. *item* yang telah didapatkan dari hasil *filtering* pertama disimpan untuk dilakukan *filtering* kedua. Dan memasuki *filtering* yang terakhir, peneliti memilih menggunakan pendekatan *item-based filtering* dengan algoritma *Singular Value Decomposition* (SVD) sebagai metode *matrix factorization* untuk memprediksi *rating item*.

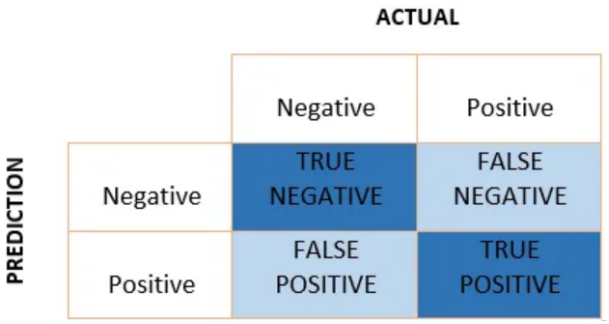
Sedangkan tahapan *hybrid filtering* pada skenario ketiga, *hybrid filtering* akan melakukan *filtering item* yang pertama menggunakan TF-IDF sebagai alat *filtering* untuk kemudian dicari kesamaan *item* menggunakan *cosine similarity*. Dari *filtering item* ini, diambil *top*-100 *item* anime dengan kemiripan antara *item* dengan query_item_id tertinggi dan disimpan sebelum dilakukan *filtering* yang terakhir. Dan memasuki *filtering* yang terakhir, peneliti memilih menggunakan pendekatan *item-based filtering* dengan algoritma *Singular Value Decomposition* (SVD) sebagai metode *matrix factorization* untuk memprediksi *rating item*.

Dari filtering terakhir pada semua skenario ini, diperoleh hasil akhir penelitian sistem *hybrid filtering* anime. Hasil penelitian sistem *hybrid filtering* anime ini berupa rekomendasi top-n *item* anime yang memiliki nilai prediksi *rating item* tertinggi. *Top-n item* anime dengan pediksi *rating* tertinggi inilah yang nantinya akan diberikan kepada *user* sebagai hasil rekomendasi dengan kriteria sesuai yang diinginkan oleh *user*.

3.3.6 Evaluasi Performansi

Pada tahap terakhir penelitian ini, dilakukan evaluasi performansi dari sistem pemberi rekomendasi *hybrid filtering* yang dibangun. Adapun metrik evaluasi yang digunakan diantaranya adalah *recall*, *precission*, dan *f1-score*. Ketiga metrik tersebut termasuk ke dalam metrik akurasi pendukung keputusan yang populer digunakan yang dapat membantu *user* dalam memilih *item* yang berkualitas tinggi dari kumpulan *item* yang tersedia. Ketiga metrik ini juga memandang prosedur prediksi sebagai operasi biner yang membedakan *item* yang memiliki kualitas bagus dari *item* dengan kualitas yang buruk [7].

Adapun untuk dapat menghitung kinerja dari ketiga metrik ini diperlukan bantuan *confusion matrix*. *Confusion matrix* merupakan matriks berukuran 2x2 yang digunakan untuk melakukan klasifikasi biner dengan *actual value* pada satu sumbu matrix dan *prediction value* pada sumbu matriks lainnya seperti yang digambarkan pada gambar (8). Pada gambar tersebut, diketahui *confusion matrix* memiliki beberapa nilai dan berikut merupakan penjelasan dari masing-masing nilai tersebut[1]:



Gambar 8. *Confusion Matrix*

- *True Negative* (TN): Baik *prediction value* dan *actual value* keduanya bernilai negatif yang artinya model sistem memprediksi kelas negatif dengan benar.
- *False Negative* (FN): *Prediction value* bernilai negatif, namun *actual value* bernilai positif yang artinya model sistem membuat prediksi yang salah dari kelas positif.
- *False Positive* (FP): *Prediction value* bernilai positif, namun *actual value* bernilai negatif yang artinya model sistem membuat prediksi yang salah dari kelas negatif.
- *True Positive* (TP): Baik *prediction value* dan *actual value* keduanya bernilai positif yang artinya model sistem memprediksi kelas positif dengan benar.

[1]https://towardsdatascience.com/performance-metrics-confusion-matrix-precision-recall-and-f1-score-a8fe076a2262, diakses pada 10 Juli 2024

Dengan adanya *prediction value* dan *actual value* dari *confusion matrix*, metrik *recall*, *precission*, dan *f1-score* dapat dihitung menggunakan persamaan (5), (6), dan (7). *Recall* adalah menghitung nilai sebagian kecil dari *item* relevan yang juga relevan bagi *user*. Sedangkan *precission* adalah menghitung nilai sebagian kecil dari *item* yang diprediksi untuk direkomendasikan sistem relevan bagi *user*. Sementara *f1-score* adalah metrik yang menggabungkan *precission* dan *recall* untuk memberikan gambaran seberapa bagus performansi dari sistem yang dibangun. *F1-score* cocok digunakan ketika sulit memutuskan dalam membandingkan berbagai model ternyata ditemukan nilai *precission* yang tinggi dan *recall* yang rendah. Dan *f1-score* dapat memberikan bobot yang sama terhadap *precission* dan *recall*, sehingga *f1-score* ini juga dapat bekerja dengan pada *dataset* yang digunakan ternyata tidak seimbang[2] [7].

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5}$$

$$Precission = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6}$$

$$F1\text{-}Score = \frac{2 * Precission * Recall}{Precission + Recall} \tag{7}$$

4. Evaluasi

4.1 Skenario Pengujian

Di akhir penelitian, dilakukan pengujian sistem rekomendasi anime menggunakan pendekatan *hybrid* yang telah dibangun oleh peneliti. Tujuan dari pengujian ini adalah untuk mengevaluasi seberapa baik kualitas kinerja model sistem yang dibangun dalam memberikan rekomendasi *item* anime kepada user. Pengujian model sistem menggunakan beberapa metrik performansi, diantaranya *precission*, *recall*, dan *f1-score*.

Adapun dalam penelitian ini, peneliti melakukan uji coba dengan membuat 3 skenario. Skenario pertama adalah skenario utama yang dibuat untuk melakukan penelitian ini yaitu membuat sistem rekomendasi menggunakan pendekatan *hybrid* menggunakan 3 *filtering*, diantaranya *k-means clustering*, TF-IDF dengan *cosine similarity*, dan *item-based filtering* dengan algoritma SVD. Dari skenario utama tersebut, hasil performansinya akan dibandingkan dengan hasil performansi dari 2 skenario uji coba lainnya yang hanya menggunakan 2 *filtering* saja. Untuk skenario kedua, 2 *filtering* yang digunakan diantaranya *k-means clustering*, dan *item-based filtering* dengan algoritma SVD. Sedangkan skenario ketiga, 2 *filtering* yang digunakan diantaranya TF-IDF dengan *cosine similarity*, dan *item-based filtering* dengan algoritma SVD.

4.2 Analisis Hasil Pengujian

Untuk mengetahui kinerja sistem yang dibangun dengan 3 skenario yang berbeda, peneliti melakukan pengujian ketiga sistem dan menilai kinerja sistem dengan menggunakan metrik performansi diantaranya *precission*, *recall*, dan *f1-score*. Dari masing-masing skenario yang dibuat, peneliti terlebih dahulu mendapatkan *top-n item recommendation* untuk masing-masing *user* dengan n sebanyak 10 item. Adapun medapatkan *top-n item recommendation* yang didapatkan dengan mempertimbangkan semua *user* mencari rekomendasi *item* yang mirip dengan masing-masing *item* yang ada. Dan tentunya dari ketiga skenario yang dibuat akan menampilkan hasil *top*-10 *item recommendation* yang berbeda-beda. Meskipun dalam beberapa kondisi tertentu, hasil dari antar skenario memiliki beberapa rekomendasi *item* yang sama. Tabel (5) merupakan contoh hasil *top*-10 *item recommendation* dari ketiga skenario untuk query_user_id 'MultiLeon' yang mencari rekomendasi *item* yang mirip dengan query_item_id '37060'. Dan *cell* yang berwarna kuning menunjukkan hasil rekomendasi beberapa *item* yang sama meski berbeda skenario.

Setelah top-n *item recommendation* didapatkan, peneliti mulai melakukan perhitungan *precission*, *recall*, dan *f1-score* untuk semua hasil *top-n item recommendation*. Untuk mempermudah analisa, peneliti hanya melihat rata-rata perbandingan dari semua hasil *precission*, *recall*, dan *f1-score* untuk masing-masing skenario. Nilai rata-rata *precission*, *recall*, dan *f1-score* dari pengujian masing-masing skenario dapat dilihat pada tabel (6). Dan untuk mempermudah perbandingan antar skenario, hasil perbandingan dapat dilihat melalui diagram batang pada gambar (9).

Dari tabel dan gambar diagram batang performansi hasil uji coba untuk masing-masing skenario tersebut, dapat dilihat bahwa secara kesuluruhan tidak satupun metrik performansi dari ketiga skenario yang mencapai nilai

[2]https://towardsdatascience.com/performance-metrics-confusion-matrix-precision-recall-and-f1-score-a8fe076a2262, diakses pada 10 Juli 2024

Tabel 5. **Contoh Hasil Top-10 *item* Recommendation**

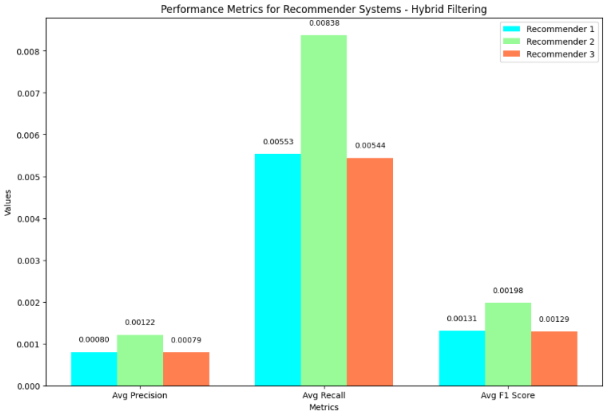
No.	query_user_id	query_item_id	item_id top-10 *item recommendation*		
			***recommender* 1**	***recommender* 2**	***recommender* 3**
1	MultiLeon	37060	9733	35503	2129
2			40363	10257	36595
3			9732	9733	11543
4			17897	33573	6856
5			36057	40363	32847
6			6513	9732	33886
7			40555	38444	37533
8			32225	19521	37357
9			32092	33106	36598
10			29453	31872	8716

Tabel 6. **Performansi Hasil Uji Coba Penelitian**

Skenario Pengujian	***Filtering* yang Digunakan Secara Berurutan**			**Metrik Performansi**		
	***Filtering* 1**	***Filtering* 2**	***Filtering* 3**	***Precission***	***Recall***	***F1-Score***
recommender 1	*K-Means Clustering*	TF-IDF ∼*Cosine Similarity*	*Item-based Filtering* ∼Algoritma SVD	0.00080	0.00553	0.00131
recommender 2	*K-Means Clustering*	*Item-based Filtering* ∼Algoritma SVD	-	0.00122	0.00838	0.00198
recommender 3	TF-IDF ∼*Cosine Similarity*	*Item-based Filtering* ∼Algoritma SVD	-	0.00079	0.00544	0.00129

sebesar 0.1. Meskipun begitu, skenario *recommender* 2 yang hanya menggunakan 2 *filtering* (*k-means clustering* dan *item-based filtering* dengan algoritma SVD) saja ternyata lebih unggul dari kedua skenario lain termasuk skenario utama dalam penelitian ini. Namun meski skenario *recommender* 2 lebih unggul daripada yang lain, tak satupun skenario pengujian yang menghasilkan nilai rata-rata *f1-score* diatas 0.5 atau mendekati 1 yang artinya performa sistem yang dibangun jauh dari kata baik. Disamping itu, terdapat ketidakseimbangan antara rata-rata *precission* dan rata-rata *recall* dari ketiga skenario. Ketiga skenario menghasilkan rata-rata *recall* yang lebih tinggi daripada rata-rata *precission* yang lebih rendah.

Analisa untuk pengujian ketiga skenario model yang dibangun bisa disimpulkan bahwa ketiga model tersebut terlalu banyak memberikan prediksi positif termasuk pada prediksi yang tidak relevan dengan user. Hal ini terjadi bisa disebabkan oleh beberapa faktor, seperti *sparsity* dan *scalability*. *Sparsity* ditemukan dalam penelitian ini sebab dari keseluruhan data yang dipakai, hanya sejumlah kecil *item* yang telah diberi rating oleh user. Hal ini menyebabkan sebagian isi matriks *item-user* kosong (*sparse*) dan menyulitkan model untuk dapat menemukan kemiripan antar *item* dengan tepat. Sedangkan scalability yang ditemukan dalam penelitian ini disebabkan oleh besarnya skala *dataset* yang digunakan sehingga model sistem yang dibangun untuk bisa menghitung kemiripan antar *item* sangat memakan banyak waktu dan menjadi tidak efisien. Meskipun di awal penelitian ini jumlah *dataset* yang digunakan sudah dikurangi banyak, ternyata *scalability* masih tidak dapat dihindari.



Gambar 9. Diagram Perbandingan Metrik Performansi Antara Ketiga Scenario *Recommender*

5. Kesimpulan

Penelitian ini telah berhasil mengimplementasikan sebuah sistem rekomendasi anime dengan pendekatan hybrid menggunakan tiga metode *filtering*, yaitu *k-means clustering*, TF-IDF dengan *cosine similarity*, dan *item-based filtering* dengan algoritma SVD. Pendekatan *hybrid* yang diterapkan menggunakan *desain pipelined hybridization* dengan strategi *hybrid cascade*, dimana 3 metode *filtering* diaplikasikan secara berurutan untuk menghasilkan rekomendasi top-n *item recommendation*.

Meskipun tujuan awal penelitian ini adalah untuk melihat apakah pendekatan *hybrid* bisa mengatasi masalah umum dalam sistem rekomendasi, seperti masalah kelangkaan data (*sparsity*) dan masalah skala (*scalability*). Ternyata hasil penelitian yang diperoleh menunjukkan bahwa masalah-masalah tersebut masih dapat ditemukan di dalam penelitian ini. Kemungkinan besar, kedua masalah tersebut disebabkan oleh ukuran dataset yang besar dan masih banyaknya data *rating* yang kosong. Atau juga bisa disebabkan karena penerapan metode *filtering* yang digunakan kurang tepat atau optimal.

Penemuan ini menunjukkan bahwa meskipun di beberapa penelitian lain pendekatan *hybrid* dapat meningkatkan kualitas rekomendasi, ternyata masih diperlukan penelitian lebih lanjut untuk mengatasi tantangan yang ada, terutama dalam menangani *dataset* yang besar dan kompleks. Pada penelitian selanjutnya, peneliti berharap dapat melanjutkan penelitian dengan mengembangkan metode yang lebih efektif untuk mengurangi masalah *sparsity* dan meningkatkan skalabilitas, sehingga sistem rekomendasi anime yang dibangun dapat berfungsi lebih baik dalam berbagai konteks penggunaan.

Daftar Pustaka

[1] M. Ago. Anime dataset with reviews - myanimelist. https://www.kaggle.com/datasets/marlesson/myanimelist-dataset-animes-profiles-reviews?select=animes.csv, 2020. Diakses pada 10 November 2022.

[2] P. Bholowalia and A. Kumar. Ebk-means: A clustering technique based on elbow method and k-means in wsn. *International Journal of Computer Applications*, 105(9), 2014.

[3] R. Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12:331–370, 2002.

[4] M. Chiny, M. Chihab, O. Bencharef, and Y. Chihab. Netflix recommendation system based on tf-idf and cosine similarity algorithms. *no. Bml*, pages 15–20, 2022.

[5] M. F. Hafidz, S. Lestari, et al. Solution to scalability and sparsity problems in collaborative filtering using k-means clustering and weight point rank (wp-rank). *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 7(4):743–750, 2023.

[6] J. A. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[7] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh. Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, 16(3):261–273, 2015.

[8] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender Systems: An Introduction*. Cambridge University Press, 2011.

[9] A. Kulkarni, A. Shivananda, A. Kulkarni, and V. A. Krishnan. *Applied Recommender Systems with Python: Build Recommender Systems with Deep Learning, NLP and Graph-Based Techniques*. Springer, 2022.

[10] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor. *Recommender Systems Handbook*. Springer, 2010.

[11] L. Rokach, B. Shapira, and F. Ricci. *Recommender systems handbook*. Springer, 2022.

[12] S. Souabi, A. Retbi, M. K. Idrissi, and S. Bennani. A recommendation approach in social learning based on k-means clustering. In *2020 International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–5. IEEE, 2020.