# BAB I

# PENDAHULUAN

## Latar Belakang

Saat ini sistem rekomendasi telah diterapkan di berbagai domain seperti musik, film, buku, dan produk (Yang, 2019). Penggunaan teknik rekomendasi yang akurat dan efisien sangat penting bagi suatu sistem untuk memberikan hasil yang baik dan bermanfaat kepada setiap penggunanya (Isinkaye, et al., 2015). Sistem rekomendasi *Collaborative Filtering* dapat merekomendasikan *item* berdasarkan kumpulan *user* dengan preferensi yang sama (Rahmawati, et al., 2018). Tetapi, sistem rekomendasi ini mengalami masalah *sparsity* pada matriks *rating*. Banyak *user* hanya menilai sejumlah kecil *item* sehingga rekomendasi yang dihasilkan memiliki akurasi yang sangat rendah (Saeed & MansooriI, 2017). Sementara dalam konteks perpustakaan digital, banyaknya koleksi dan kebutuhan informasi yang beragam membuat *rating* dari *user* sangat diperlukan. Tetapi, *rating* pada perpustakaan digital cenderung jarang diberikan oleh *user* (Wenige & Ruhland, 2018). Oleh karena itu, diperlukan suatu teknik yang dapat memprediksi *rating* *user* terhadap *item* (Bobadilla, et al., 2020).

*Deep Learning*, sebagai jenis pendekatan *machine learning* telah berhasil diterapkan di banyak domain penelitian berbeda, seperti *computer vision*, *speech recognition*, *natural language processing* dan lain sebagainya (Liu & Wu, 2017). Menggunakan *Deep Learning* pada sistem rekomendasi memungkinkan model secara otomatis mempelajari fitur yang digeneralisasi dengan baik bagi *user* dan *item* dari sumber daya yang berbeda. Dengan menggunakan *Deep Learning* dalam pemodelan berbagai tipe data, sistem rekomendasi akandapat lebih memahami apa yang dibutuhkan *user* dan hal tersebut akan meningkatkan hasil rekomendasi (Ikasari, et al., 2018).

Model *Deep Learning* untuk *Collaborative Filtering* merupakan penggabungan *Collaborative Filtering* berbasis *Matrix Factorization* dengan algoritma *Deep Learning*. *Deep Collaborative Filtering* yang merupakan model *hybrid*, menggunakan matriks *rating* dan *side information* serta menjembatani *Matrix Factorization* dan fitur *learning*. *Deep Collaborative Filtering* mengintegrasikan *Matrix Factorization* dan fitur *Deep Learning* dengan memodelkan pemetaan antara *latent factors* yang digunakan dalam *Collaborative Filtering* dan *latent layers* dalam *Deep models* (Li, et al., 2015).

Penerapan *Deep Collaborative Filtering* pada perpustakaan digital akan memberikan rekomendasi *item* lebih akurat daripada hanya menggunakan *Collaborative Filtering*. *User* pada perpustakaan digital jarang memberikan *rating* pada *item* (Wenige & Ruhland, 2018). Jika hanya menggunakan *Collaborative Filtering*, hasil rekomendasi tidak akan akurat karena terjadi *sparsity* matriks *rating*. Oleh karena itu, *Deep Collaborative Filtering* diharapkan dapat mengatasi masalah tersebut pada aplikasi perpustakaan digital.

Berdasarkan uraian di atas, maka dikembangkan sebuah aplikasi yang dapat memberikan rekomendasi kepada *user* tanpa mengalami *sparsity* pada matriks *rating* dan dituangkan dalam tugas akhir dengan judul “APLIKASI REKOMENDASI PERPUSTAKAAN DIGITAL MENGGUNAKAN METODE DEEP COLLABORATIVE FILTERING BERBASIS MOBILE DAN WEB”.

## Rumusan Masalah

Berikut rincian permasalahan yang telah dituliskan pada latar belakang:

1. Bagaimana penerapan dari sistem rekomendasi menggunakan metode *Deep Collaborative Filtering* pada perpustakaan digital?
2. Bagaimana metode *Deep Collaborative Filtering* mengatasi *sparsity* matriks *rating* pada perpustakaan digital?
3. Bagaimana *user* perpustakaan digital mendapatkan *item* yang sesuai dengan kriteria dan keinginannya?

## Tujuan

Tujuan tugas akhir ini yaitu sebagai berikut:

1. Menerapkan *Deep* *Collaborative Filtering* pada perpustakaan digital.
2. Mengatasi *sparsity* matriks *rating* pada perpustakaan digital.
3. Mengembangkan aplikasi perpustakaan digital yang memberikan rekomendasi dalam menentukan *item* yang sesuai dengan kebutuhan dan keinginan *user*.

## Manfaat

Manfaat yang diharapkan dari tugas akhir ini adalah sebagai berikut:

1. Aplikasi perpustakaan digital yang bersifat *easy to use*, sehingga mempermudah dalam melakukan pencarian dan menampilkan hasil sesuai keinginan *user*.
2. Aplikasi ini dapat membantu masyarakat untuk terus membaca buku dimana saja dan kapan saja.

## Batasan Masalah

Batasan masalah pada penelitian implementasi *Deep Collaborative Filtering* untuk menyelesaikan rekomendasi pada perpustakaan digital dalam mendapatkan *item* sesuai dengan kebutuhan dan keinginan user sesuai dengan identifikasi dan rumusan masalah. Pembatasan masalah yang digunakan pada penelitian ini adalah:

1. *Rating user* digunakan sebagai parameter dalam sistem rekomendasi *Deep Collaborative Filtering* dengan menggunakan *history* pencarian dan peminjaman yang dilakukan oleh *user* dengan asumsi ketika *user* telah meminjam *item* perpustakaan berarti *user* telah memberikan *rating* pada *item* tersebut.
2. Hanya menggunakan teknik *Deep Collaborative Filtering* karena pada sistem rekomendasi ini *item* yang pernah dicari atau dipinjam *user* akan menjadi patokan untuk pemberian rekomendasi.
3. *Mobile application* sebagai produk dari Tugas Akhir ini hanya dapat dipasang pada *smartphone* dengan *operating system* Android versi Lollipop 5.0 dan belum tersedia dalam sistem operasi IOS.
4. Jumlah aktor dalam sistem ini yaitu dari 3(tiga), terdiri dari admin, member dan *user* (aktor yang belum mendaftar ke dalam sistem) dengan masing-masing aktor yaitu sebagai berikut:

Admin:

1. Mengelola data item digital.
2. Mengelola pinjaman *item* digital.
3. Melakukan verifikasi member terdaftar.
4. Mengelola laporan.

Member:

1. Mendapat rekomendasi *item* digital dari algoritma *Deep Collaborative Filtering*.
2. Mencari, meminjam, membaca, mengembalikan dan memperpanjang *item* digital.
3. Memberi *rating* dan *review* *item* digital yang telah dipinjam.

*User*:

1. Mendapat rekomendasi *item* digital dari algoritma *Deep Collaborative Filtering*.
2. Mencari dan melihat rincian *item* digital.
3. Melakukan pendaftaran pada sistem.

## Metodologi Pengembangan Sistem

Metodologi yang digunakan pada pengembangan sistem ini yaitu dengan model *waterfall*. Metode *waterfall* merupakan metodologi pengembangan sistem yang bersifat sistematis sehingga dapat menghasilkan software yang baik dan penataan perpustakaan yang terurut dengan rapi karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan.

1. Analisa Proses

Pada tahapan ini perhitungan untuk algoritma *Deep Collaborative Filtering* yang akan digunakan dengan contoh kasus atau angka sederhana disajikan secara tahap demi tahap serta menggunakan *flowchart* untuk menganalisis proses algoritma yang digunakan*.*

1. Studi Literatur

Pada tahapan ini adalah mencari literatur yang berhubungan dengan topik yang diangkat. Tujuan dari tahapan ini adalah penulis memahami konsep, metode dan teknologi Native.

1. Data *Processing*

Pada tahap ini akan dilakukan pengolahan data dengan baik, dimana akan melakukan transformasi data ke suatu format yang prosesnya lebih mudah dan efektif sebagai kebutuhan pengguna. Tahapan *processing* ini terdiri dari pemilihan atribut, penggabungan data, *data cleaning*, dan standarisasi data.

1. Perancangan

Pada tahapan ini adalah tahapan lanjutan dari tahapan analisis dengan melakukan perancangan tampilan (*interface*) menggunakan Figma, pemodelan basis data menggunakan Microsoft Visio untuk perancangan ERD *(Entity Relationship Diagrams).*

1. Implementasi

Pada tahapan ini dilakukan penulisan kode program, untuk *website* dengan bahasa *Javascript* dan *Framework Laravel,* untuk aplikasi *mobile* menggunakan bahasa Kotlin dan dataset dengan [https://www.kaggle.com/](https://www.kaggle.com/jealousleopard/goodreadsbooks).

1. Pengujian

Pengujian software diperlukan untuk memastikan aplikasi yang dibangun dapat berjalan sesuai fungsionalitasnya.

1. Pengujian sistem rekomendasi *Deep Collaborative Filtering* untuk mengukur keakuratan nilai rekomendasi dimana nilai dari hasil pengujian pada sistem rekomendasi akan menjadi nilai parameter akurat atau tidaknya rekomendasi yang diberikan kepada *user*.
2. Pengujian fungsionalitas dari aplikasi perpustakaan, menggunakan metode *Black Box Testing* untuk menguji perangkat lunak apakah hasilnya sesuai dengan yang diharapkan.
3. Kesimpulan hasil pengujian

Penarikan Kesimpulan yang diambil berdasarkan hasil pengujian yang dilakukan dengan metode *Deep Collaborative Filtering.*

# BAB II

# TINJAUAN PUSTAKA

## Perpustakaan Digital

Perpustakaan Digital didefinisikan sebagai kumpulan informasi elektronik yang berisi repository objek digital yang besar dan beragam, dan dapat diakses oleh sejumlah besar pengguna yang tersebar secara geografis. Objek digital yang dimaksud seperti teks, gambar, peta, suara, video, katalog, dan jurnal ilmiah (Khiste, et al., 2018). Setiap *user* dapat mengakses sumber daya hingga informasi yang tersedia di perpustakaan dengan menggunakan komputer dan koneksi ke jaringan perpustakaan tanpa harus hadir secara fisik di perpustakaan (Nahak & Padhi, 2019)

Pada umumnya, Perpustakaan Digital melayani mesin, manajer informasi, dan pemakai informasi. Semuanya ini demi mendukung manajemen koleksi, menyimpan, pelayanan bantuan, dan penelusuran informasi. Perpustakaan Digital dengan konsep *resource* c*enter* atau teknologi penyimpanan elektronik dapat mengurangi ruang penyimpanan buku, jurnal, dan majalah yang masih berbasis kertas. Konsep r*esource* *center* juga mendukung fasilitas pencarian pada database sehingga memberikan kemudahan dalam pencarian (Aithal, 2016).

### Klasifikasi Buku pada Perpustakaan

Klasifikasi merupakan salah satu alat tertua yang dikembangkan oleh ahli pustakawan dan dianggap sebagai dasar dari kepustakawanan karena fungsinya yang serba guna. Tujuan tradisional klasifikasi untuk menempatkan objek fisik di rak telah kehilangan fungsinya dalam kasus sumber daya elektronik jarak jauh. Namun, klasifikasi memiliki fungsi lain dalam kategorisasi, yang lebih berguna dalam kasus sumber daya elektronik, karena berperan dalam pengorganisasian subjek (Rikowski, 2012). Berikut beberapa sistem klasifikasi yang digunakan sebagai pedoman dalam perpustakaan, yaitu:

1. Dewey Decimal Classification

Dewey Decimal Classification atau disingkat DDC merupakan sistem yang mengorganisasi pengetahuan secara umum, yang terus menerus direvisi untuk mengikuti perkembangan pengetahuan. DDC diterbitkan pertama kali pada tahun 1873 oleh Melvin Dewey. Edisi termutakhir dari DDC adalah DDC 23 yang diterbitkan pada tahun 2011 oleh Joan S. Mitchell dengan hak cipta pada Online Computer Library Centre, Inc. (OCLC) (Suharyanto, 2012).

Tabel : Kelas Umum Dewey Decimal Classification 23

Sumber (Suresha & Narayanaswamy, 2016)

|  |  |  |
| --- | --- | --- |
| No | Kode | Subject |
| 1 | 000 | *Computers, Information and General Reference* |
| 2 | 100 | *Philosophy and Psychology* |
| 3 | 200 | *Religion* |
| 4 | 300 | *Social Sciences* |
| 5 | 400 | *Language* |
| 6 | 500 | *Science and Mathematics* |
| 7 | 600 | *Technology* |
| 8 | 700 | *Arts and Recreation* |
| 9 | 800 | *Literature* |
| 10 | 900 | *History and Geography* |

1. Universal Decimal Classification

Universal Decimal Classification atau disingkat UDC merupakan klasifikasi internasional pertama yang dirancang untuk informasi bibliografi dan pekerjaan dokumentasi yang disebut perluasan dari Dewey Decimal Classification. Universal Decimal Classification mencakup seluruh domain pengetahuan, dengan 10 kelas utama dan kelas utama ke-4 masih kosong sebagai berikut:

Tabel : Kelas Utama Universal Decimal Classification

Sumber (Satija, 2008)

|  |  |  |
| --- | --- | --- |
| No | Kode | Subject |
| 1 | 0 | *Generalities* |
| 2 | 1 | *Philosophy; Psychology* |
| 3 | 2 | *Religion, Theology* |
| 4 | 3 | *Social Sciences* |
| 5 | 4 | *[Vacant]* |
| 6 | 5 | *Natural Sciences; Mathematics* |
| 7 | 6 | *Technology* |
| 8 | 7 | *The Arts* |
| 9 | 8 | *Language; Linguistics; Literature* |
| 10 | 9 | *Geography; Biography; History* |

Dari ke-10 kelas utama tersebut dapat diperluas lagi dengan notasi dari table bantu yang terdiri dari dua jenis, yaitu Alat bantu umum yang dapat diterapkan secara universal dan Alat bantu khusus yang dapat diterapkan secara terbatas atau secara local (Satija, 2008).

## Sistem Rekomendasi

Sistem rekomendasi merupakan fitur yang berfungsi untuk membantu *user* untuk menemukan saran tentang *item* apa yang sebaiknya digunakan atau dipilih. Secara umum Sistem Rekomendasi ditujukan untuk individu yang kekurangan pengalaman atau kompetensi yang cukup untuk mengevaluasi banyaknya jumlah alternatif *item* yang ada pada suatu kasus tertentu (Elahi, 2016). Terdapat beragam teori yang dapat digunakan untuk membuat sistem rekomendasi seperti *Bayesian Algorithm* (Bobadilla, 2017)*, Decision Tree Algorithm* (Guabassi, 2016) dan *Matrix factorization-based* (Aleksandrova, 2017) dan lain-lain. Contoh *Matrix factorization-based* dapat dilihat pada gambar 2.1.



Gambar 2. 1 Contoh *Matrix factorization-based*

(Sumber : <https://developers.google.com/machine-learning/recommendation/collaborative/matrix>)

### Input data dalam Sistem Rekomendasi

Dalam melakukan pengumpulan data *user* untuk sistem rekomendasi, terdapat dua cara (Zulkarnain, 2013) yaitu dengan cara Eksplisit dan Implisit. Eksplisit merupakan pengumpulan data yang diperoleh berdasarkan *feedback* dari pengguna secara langsung, data dapat diperoleh dengan meminta pendapat langsung dari pengguna, bisa berupa rating, *likes/dislikes*, maupun kata pencarian (*keyword*). Metode Implisit merupakan pengumpulan data berdasarkan pengamatan pola kecenderungan pengguna, seperti kecenderungan *user* terhadap suatu *item* dengan kriteria tertentu yang dimiliki *item* tersebut. Metode pengumpulan data pada teknik implisit tidak melibatkan kontribusi pengguna melainkan dengan cara pendekatan dari perilaku pengguna dan menyimpulkan data-data yang sudah dikumpulkan oleh sistem. Berdasarkan pendekatan yang digunakan untuk menghasilkan rekomendasi, sistem rekomendasi dibagi menjadi enam bagian (Francesco Ricci, 2011) yaitu:

1. Content Based System

Dalam metode content-based, rekomendasi dibuat berdasarkan kemiripan atribut-atribut dari suatu *item* dengan preferensi pengguna tersebut (Candillier, et al., 2009) .Misalnya seseorang mempunyai kecenderungan suka dengan membaca buku algoritma, maka sistem akan merekomendasikan buku tentang pemrograman.

1. Collaborative Filtering System

*Collaborative filtering* menghasilkan rekomendasi *item* berdasarkan histori pengguna lain yang memiliki selera sama dengan pengguna tersebut (Candillier, et al., 2009). Sistem akan memprediksi ketertarikan pengguna tersebut terhadap *item* yang akan direkomendasikan berdasarkan data pengguna lain yang memiliki ketertarikan yang sama.

1. Demographic System

Jenis sistem rekomendasi ini akan merekomendasikan *item* berdasarkan demografi profil pengguna. Misalnya sistem akan merekomendasikan *item* berdasarkan negara/daerah asal dari pengguna, sehingga *item* yang direkomendasikan akan berbeda tergantung dari demografi pengguna yang dituju.

1. Knowledge-based System

Sistem akan merekomendasikan *item* berdasarkan ranah pengetahuan yang spesifik tentang bagaimana fitur *item* tertentu memenuhi kebutuhan dan preferensi pengguna, dan bagaimana *item* tersebut berguna bagi pengguna. Sebagai contoh sistem akan merekomendasikan komputer dengan spesifikasi tinggi bagi pengguna dengan preferensi gamer, dan akan merekomendasikan komputer dengan spesifikasi rendah bagi pengguna dengan kebutuhan biasa saja.

1. Community-based System

Jenis sistem rekomendasi ini memodelkan dan memperoleh informasi tentang hubungan sosial dari pengguna dan preferensi pengguna lain yang berhubungan dengan pengguna tersebut. Rekomendasi yang diberikan berdasarkan rating yang diberikan oleh pengguna lain tersebut.

1. Hybrid System

Jenis sistem rekomendasi hybrid merupakan gabungan dari metode-metode yang telah disebutkan. Sistem rekomendasi *hybrid* akan menggunakan lebih dari satu metode untuk mentolerir kekurangan dari metode lainnya.

Dari beberapa metode atau teknik yang digunakan dalam sistem rekomendasi. Setiap metode disesuaikan dengan permasalahan dalam menghasilkan sebuah informasi yang sesuai. Penggabungan algoritma *Collaborative Filtering* dengan *Deep Learning* dapat memberikan pendekatan yang efektif (Li, et al., 2015). Oleh karena itu, pada tugas akhir ini kami menggunakan metode *Collaborative Filtering System*.

## Collaborative Filtering

*Collaborative Filtering* merupakan pemfilteran kolaboratif yang relatif sederhana, efektif, dan telah digunakan secara luas oleh banyak situs web komersial. *Collaborative Filtering* memanfaatkan data *rating* *item* yang ada oleh *user* untuk membuat prediksi tentang preferensi *user*. Pendekatan *Collaborative Filtering* dapat dibagi menjadi dua kelompok: Pendekatan berbasis memori dan model. Pendekatan berbasis memori (berbasis *heuristik*) membuat prediksi berdasarkan kesamaan antara *user* dan *item*. Pendekatan berbasis model berusaha membuat model prediksi melalui *machine learning*. Secara khusus, model berbasis faktorisasi matriks telah memperoleh popularitas karena memiliki akurasi dan skalabilitas yang relatif tinggi. Meski demikian, *Collaborative Filtering* masih memiliki tantangan terhadap *sparsity* matriks *rating* (Bo Yang, 2016).

*Collaborative Filtering* bertujuan untuk merekomendasikan *item* baru atau memperkirakan kegunaan *item* tertentu terhadap *user* tertentu berdasar dari kesukaan *user* di masa lalu dan pandangan dari *user* lain yang berpikiran sama. Terdapat dua tugas yang dapat dilakukan oleh *Collaborative Filtering* yakni:

1. *Rating* *Prediction* yaitu memprediksi peringkat yang akan dimiliki oleh *item* yang tak terlihat kepada *user* target
2. *Recommendation Task* yaitu menyediakan daftar rekomendasi *top*-N dari *item* relevan yang tak terlihat kepada *user* target

*Collaborative Filtering* beroperasi di dalam sebuah *matrix rating*. *Rating* yang diberikan oleh *user* terhadap *item* direpresentasikan sebagai R dan nilai *rating* tersebut merupakan bilangan bulat tidak negatif atau bilangan real dengan jarak tertentu. *Collaborative Filtering* mencoba memprediksi *rating* yang akan diberikan oleh *user* terhadap suatu *item* yang belum pernah di beri *rating* sebelumnya.

Misalnya, terdapat lima *user* 𝑢1,2,𝑢3,..𝑢5 dan lima *item* 𝑖1,2,𝑖3,..𝑖5. Sebuah sistem rekomendasi ingin memprediksi berapa *rating* yang akan diberikan oleh 𝑢1 kepada 𝑖5.

Table 2.1 Matrix *User* x *Item* pada Sistem Rekomendasi *Collaborative Filtering*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *Item* | *Item* | *Item* | *Item* | *Item* |
| *User* | 5 | 7 | 5 | 7 | 7 |
| *User* | 5 | 7 | 5 | 7 | 9 |
| *User* | 5 | 7 | 5 | 7 | 9 |
| *User* | 6 | 6 | 6 | 6 | 5 |
| *User* | 6 | 6 | 6 | 6 | 5 |

Terdapat sebuah cara yang sering digunakan dalam menghitung atau memprediksi *rating user u* terhadap *item*

*i* dengan mengkalkulasikan rata-rata *rating user* lain (*neighbor*) **Invalid source specified.**, yaitu rumus *weighted sum*:

Dimana:

(u,𝑖) adalah prediksi *rating* *user* u terhadap *item* i.

(𝑢′,) adalah *rating* *user* u’ (*user* selain u) terhadap *item* i.

𝑠(𝑢,𝑢′) adalah kemiripan antara *user* 𝑢 dengan 𝑢′.

N(u) adalah satu set *user* lain yang mirip dengan *user* u.

*Cosine-based Similarity* adalah cara terbaik untuk menghitung kemiripan dua *user* **Invalid source specified.**. *Cosine-based similarity* dapat dikalkulasikan dengan formula sebagai berikut :

Dimana:

𝑠𝑖𝑚 (𝑢,’) adalah kemiripan antara *user* 𝑢 dengan 𝑢’.

(𝑢,’) adalah satu set *item* yang telah di-*rating* oleh kedua *user* 𝑢 dengan 𝑢’.

Sistem rekomendasi *Collaborative Filtering* memiliki beberapa kekurangan, diantaranya adalah:

1. Jika terdapat *item* baru di dalam sistem tersebut, maka sistem tidak dapat merekomendasikan *item* tersebut sampai *user* lain berminat pada *item* tersebut. (*First Rater Problem*)
2. Hasil rekomendasi tidak selamanya handal. Terdapat kemungkinan adanya sebuah *user* yang hanya memiliki sedikit neighbor dengan tingkat kesamaan (similarity) yang kecil. (*Grey Sheep Problem*)
3. Hasil rekomendasi yang tidak handal yang dikarenakan matrix *rating* yang jarang. (*Sparsity Problem*)

## Matrix Factorization

Pada penelitian ini, data yang didapatkan akan direpresentasikan berbentuk matriks. Matriks tersebut ditulis dalam bentuk *Rmxn*dimana *m* merupakan banyaknya baris dan *n* merupakan banyaknya kolom*.* Dimana pada matriks tersebut setiap baris merupakan *user* dan tiap kolom merupakan *item* dan hubungan antar *user* dan *item* (*rij*) adalah *rating* yang diberikan *user* terhadap *item*.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| # | Item-1 | Item-2 | Item-3 | Item-4 | … | Item-n |
| User-1 | r11 | r12 | r13 | r14 | … | r1n |
| User-2 | r21 | r22 | r23 | r24 | … | r2n |
| User-3 | R31 | r32 | r33 | r34 | … | r3n |
| … | … | … | … | … | … | … |
| User-m | rm1 | rm2 | rm3 | rm4 | … | rmn |

Table 2.1 - Contoh matriks *rating user* terhadap suatu *item*

Pada tabel 2.1 akan terdapat nilai *rating* yang kosong. Hal ini dikarenakan *user* tidak memberikan *rating* kepada *item* yang ada. Banyaknya *user* yang hanya menilai sejumlah *item* saja menyebabkan terjadinya *sparsity* matriks *rating* (Zhang & Chow, 2016). *Matrix Factorization* adalah pendekatan *Collaborative Filtering* yang paling efektif. Ini memungkinkan kita untuk menemukan *latent factor* interaksi *user-item* dengan memfaktorkan matriks interaksi ke dalam *latent space* fitur *user-item*. Salah satu metode *Matrix Factorization* klasik adalah *Probabilistic Matrix Factorization* (PMF). Banyak algoritma yang telah dikembangkan untuk meningkatkan kinerja PMF, dengan memasukkan *side information* seperti hubungan sosial. Tetapi metode *Matrix Factorization* mengalami masalah *cold-start*, yaitu rekomendasi apa yang harus dibuat ketika *user* / *item* baru tiba di sistem. Masalah lain yang sering muncul di banyak aplikasi dunia nyata adalah ketersebaran data atau cakupan yang berkurang. Memasukkan *side information* telah menunjukkan kinerja yang menjanjikan dalam *Collaborative Filtering*. Tetapi akan bermasalah jika *side information* tidak lengkap. Oleh karena itu, fitur pembelajaran untuk *Matrix Factorization* sangat penting untuk diterapkan (Li, et al., 2015).

## Deep Collaborative Filtering

Algoritma *Deep Collaborative Filtering* adalah algoritma yang dibuat untuk mengatasi masalah *sparsity* matriks *rating* pada algoritma *Collaborative Filtering.* Penyelesaian permasalahan tersebut dilakukan dengan mempelajari *side information* yang diperoleh dari profil *user* / *item* seperti demografi *user*, genre buku, dll. *Deep Collaborative Filtering* merupakan algoritma yang dikenal sebagai penggabungan *matrix factorization* pada *Based Collaborative Filtering* dengan *Deep Learning*. *Deep Collaborative Filtering* mengintegrasikan faktorisasi matriks dan pembelajaran *deep feature* yang memodelkan pemetaan antara faktor laten yang digunakan dalam *Collaborative Filtering* dan lapisan laten dalam *deep models* (Li, et al., 2015).



Gambar 2.2 Ilustrasi penerapan kerangka *Deep collaborative Filtering*

Sumber (Li, et al., 2015)

|  |  |
| --- | --- |
| Notasi | Deskripsi |
| m | Jumlah *user* |
| n | Jumlah *item* |
| d | Dimensi *latent factor* |
| p | Dimensi *user feature* |
| q | Dimensi *item feature* |
| *R* € | Matriks *Rating* |
| *U* € | *Latent factor user* |
| *V* € | *Latent factor item* |
| *X* € | *Side information of user* |
| *Y* € | *Side information of item* |
| *W1*  € | *Mapping Function* untuk X di *auto-encoder* |
| *P1* € | *Projection matrix* untuk *U* |

Table 2.2 – Ringkasan Notasi

### Mempelajari latent factor dari rating dan side information

*Deep Collaborative Filtering* adalah model *hybrid*, yang menggunakan matriks *rating* dan *side information* serta menjembatani faktorisasi matriks dan pembelajaran fitur. Diberikan matriks rating user-*item*, *side information user* dan *side information item*, *Deep Collaborative Filtering* mendekomposisi matriks *rating* dan mempelajari *latent factor* dari *rating* dan *side information* dengan rumus berikut:

Ada dua komponen kunci dalam *Deep Collaborative Filtering*: (i) fungsi *l (R, U, V)* untuk menguraikan matriks *rating* menjadi dua matriks laten; (ii) fungsi *L (X, U)* dan *L (Y, V)* yang menghubungkan fitur kontekstual *user*/*item* dengan faktor laten. Komponen pertama yang diturunkan melalui matrix factorization, mengekstrak pengetahuan laten dari matriks *rating*. Komponen kedua yang dirancang dengan menggunakan model pembelajaran mendalam membangun koneksi *side information* dengan faktor laten (Li, et al., 2015).

### Normalisasi Data Pada Deep Collaborative Filtering

Normalisasi data dilakukan pada matriks *rating* yang mengindikasikan penilaian oleh *user* terhadap *item*. Normalisasi dilakukan dengan menggunakan  *zero-mean normalization*. Nilai *rating* dari dataset dalam skala 1 – 10 akan dinormalisasikan menjadi *zero-mean*. U1-U10 adalah *user* dan J1-J10 adalah *item* yang dinilai ataupun yang tidak dinilai oleh *user*.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** | **J7** | **J8** | **J9** | **J10** |
| **U1** | 8 | 4 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **U2** | 0 | 1 | 0 | 1 | 2 | 1 | 0 | 2 | 0 | 5 |
| **U3** | 5 | 1 | 1 | 0 | 1 | 3 | 0 | 1 | 4 | 0 |
| **U4** | 0 | 0 | 3 | 4 | 0 | 7 | 0 | 6 | 0 | 6 |
| **U5** | 0 | 0 | 1 | 0 | 0 | 5 | 2 | 1 | 3 | 0 |
| **U6** | 0 | 1 | 0 | 5 | 3 | 4 | 0 | 4 | 8 | 2 |
| **U7** | 1 | 2 | 0 | 0 | 0 | 4 | 3 | 3 | 4 | 5 |
| **U8** | 0 | 4 | 3 | 2 | 3 | 0 | 2 | 0 | 0 | 0 |
| **U9** | 0 | 3 | 4 | 2 | 0 | 1 | 2 | 2 | 1 | 6 |
| **U10** | 1 | 4 | 0 | 3 | 1 | 3 | 0 | 3 | 5 | 0 |

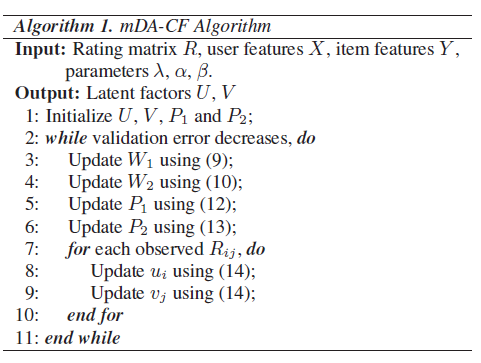
Table 2.3 – Data Sebelum Normalisasi

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** | **J7** | **J8** | **J9** | **J10** |
| **U1** | 2,989 | 1,042 | 0,555 | -0,905 | -0,905 | -0,419 | -0,905 | -0,905 | -0,905 | -0,905 |
| **U2** | -0,905 | -0,419 | -0,905 | -0,419 | 0,068 | -0,419 | -0,905 | 0,068 | -0,905 | 1,528 |
| **U3** | 1,528 | -0,419 | -0,419 | -0,905 | -0,419 | 0,555 | -0,905 | -0,419 | 1,042 | -0,905 |
| **U4** | -0,905 | -0,905 | 0,555 | 1,042 | -0,905 | 2,502 | -0,905 | 2,015 | -0,905 | 2,015 |
| **U5** | -0,905 | -0,905 | -0,419 | -0,905 | -0,905 | 1,528 | 0,068 | -0,419 | 0,555 | -0,905 |
| **U6** | -0,905 | -0,419 | -0,905 | 1,528 | 0,555 | 1,042 | -0,905 | 1,042 | 2,989 | 0,068 |
| **U7** | -0,419 | 0,068 | -0,905 | -0,905 | -0,905 | 1,042 | 0,555 | 0,555 | 1,042 | 1,528 |
| **U8** | -0,905 | 1,042 | 0,555 | 0,068 | 0,555 | -0,905 | 0,068 | -0,905 | -0,905 | -0,905 |
| **U9** | -0,905 | 0,555 | 1,042 | 0,068 | -0,905 | -0,419 | 0,068 | 0,068 | -0,419 | 2,015 |
| **U10** | -0,419 | 1,042 | -0,905 | 0,555 | -0,419 | 0,555 | -0,905 | 0,555 | 1,528 | -0,905 |

Table 2.4 – Data Setelah Normalisasi

### Mendapatkan Latent Factor dari User dan Item

*Latent factor* telah menjadi teknik yang sukses digunakan untuk membangun sistem rekomendasi. Meskipun kuncinya adalah minat pengguna secara efektif, sebagian besar penelitian difokuskan untuk mempelajari l*atent factor* dalam situasi *cold-start* dan *data sparsity* (Rao, et al., 2017). Berikut Algoritma untuk mendapatkan *latent factor user* dan *Item*.



Dimana rumus di dalam Algoritma :

a. Update W1



b. Update W2





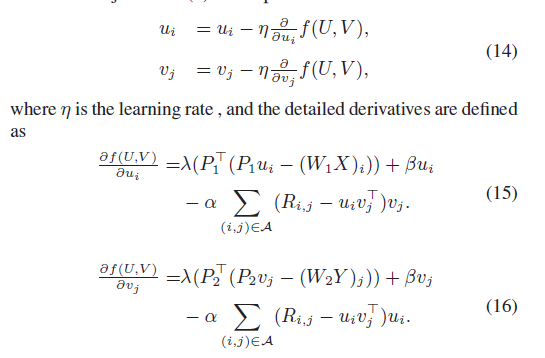
c. Update P1



d. Update P2



e. Update ui and uj



## Pengujian Perangkat Lunak dan Pengujian Keakuratan Hasil Rekomendasi

Pengujian atau testing sendiri merupakan elemen kritis dari jaminan kualitas perangkat lunak dan merupakan bagian yang tidak terpisah dari siklus hidup pengembangan software seperti halnya analisis, desain, dan pengkodean (Shi, 2010). Pengujian *software* haruslah dilakukan dalam proses rekayasa perangkat lunak atau *software engineering.* Ada beberapa jenis pengujian perangkat lunak, antara lain (Khan, 2011):

1. Pengujian Perangkat Lunak dengan *Black Box Testing*

Pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, *tester* dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengujian pada spesifikasi fungsional program. *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. *Tester* dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengujian pada spesifikasi fungsional program. *Black Box Testing* bukanlah solusi alternatif dari *White Box* *Testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing*. *Black Box Testing* cenderung untuk menemukan hal-hal yaitu sebagai berikut.

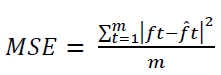
1. Fungsi yang tidak benar atau tidak ada
2. Kesalahan antarmuka (*interface errors*)
3. Kesalahan pada struktur data dan akses basis data
4. Kesalahan performansi (*performance errors*)
5. Kesalahan inisialisasi dan terminasi
6. Pengujian Keakuratan Hasil Rekomendasi

Pengujian digunakan untuk mengetahui seberapa akurat hasil rekomendasi yang diberikan sistem kepada *user*. Beberapa cara yang dapat dilakukan untuk mengevaluasi sistem rekomendasi yaitu dengan menghitung tingkat error pada hasil rekomendasi dan juga menghitung nilai presisi dari hasil *User Acceptance Test* (UAT) yang diberikan *user*.

Dalam sistem rekomendasi, hasil rekomendasi yang diberikan perlu dilakukan pengukuran untuk mengukur tingkat akurasi hasil rekomendasi dapat dilakukan dengan melihat nilai *error* pada hasil rekomendasi yang diberikan. Terdapat banyak metode perhitungan *error* seperti, *Mean Square Error* (MSE), Mean Absolute Percentage Error (MAPE) dan *Mean Absolute Error (MAE)*

1. *Mean Square Error* (MSE)

MSE adalah rata-rata absolut dari kesalahan peramalan yang dikuadratkan. Persamaan dari MSE yaitu sebagai berikut :



Dimana,

𝑓𝑡 = Permintaan aktual periode *t*

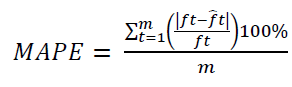
𝑓̂ = Ramalan permintaan periode *t*

m = Jumlah periode peramalan

MSE biasa dipakai untuk menghitung hasil dari peramalan namun MSE juga dapat diterapkan pada perhitungan hasil rekomendasi yaitu dengan mengganti *ft* (permintaan aktual periode *t*) menjadi *rating user* terhadap *item t* lalu (ramalan permintaan periode *t*) menjadi prediksi *rating user* terhadap *item* t dan m menjadi jumlah *item* yang dihitung.

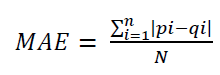
1. *Mean Absolute Percentage* (MAPE)

MAPE adalah rata-rata persentase absolut dari kesalahan peramalan dengan menghitung *error absolute* tiap periode. *Error* ini kemudian dibagi dengan *n*. Persamaan dari MAPE ini yaitu sebagai berikut



1. *Mean Absolute Error* (MAE)

Dari ketiga jenis perhitungan *error*, MAE merupakan persamaan perhitungan nilai *error* yang paling sering digunakan untuk menghitung nilai *error* dari hasil rekomendasi. *Mean Absolute Error* (MAE) merupakan persamaan yang termasuk jenis *statistical accuracy metrics* dimana MAE akan menghitung nilai rata-rata selisih antara nilai prediksi dengan nilai yang sebenarnya. Persamaan MAE yaitu sebagai berikut:



Dimana*,*

*MAE =* Nilai rata-rata kesalahan hitungan

N = jumlah *item* yang dihitung

*pi* = Nilai prediksi *item* ke i

*qi* = Nilai *rating* sebenarnya *item* ke i

Semakin rendah nilai MAE yang didapat, maka sistem semakin akurat dalam memprediksi.

# DAFTAR PUSTAKA

Aithal, D. P. S., 2016. SMART LIBRARY MODELS FOR FUTURE GENERATIONS.

Aleksandrova, M. e. a., 2017. Identifying representative users in matrix factorization-based recommender systems: application to solving the content-less new item cold-start problem. *J Intell Inf Syst.*

Bo Yang, Y. L. J. L. a. W. L., 2016. Social Collaborative Filtering by Trust. p. 1.

Bobadilla, J., Alonso, S. & Hernando, A., 2020. Deep Learning Architecture for Collaborative Filtering Recommender Systems. p. 1.

Bobadilla, J. e. a., 2017. Recommender Systems Clustering Using BayesianNon Negative Matrix Factorization. *Digital Object Identifier 10.1109/ACCESS.2017.2788138.*

Candillier, L., Jack, K., Fessant, F. & Meyer, F., 2009. State of the Art Recommender Systems. *Research Gate.*

Chen, J. et al., 2020. Deep attention user-based collaborative filtering for recommendation. p. 2.

Elahi, M., 2016. A survey of active learning in collaborative filtering recommender systems. *C O M P U T E R S C I E N C E R E V I E W2 0 ( 2 0 1 6 ).*

Francesco Ricci, L. R. B. S., 2011. Recommender System Handbook. *Kantor, P. B..*

Guabassi, I. E., 2016. Recommender system for ubiquitous learning based on decision tree. *2016 4th IEEE International Colloquium on Information Science and Technology (CiSt).*

Ikasari, D., Suhendra, A. & Farida, N., 2018. Metode Deep Learning Pada Sistem Rekomendasi : Review Paper. Volume 2, p. 47.

Isinkaye, F., Folajimi, Y. & Ojokoh, B., 2015. Recommendation systems: Principles, methods and. p. 4.

Khan, M. E., 2011. DIFFERENT APPROACHES TO BLACK BOX TESTING TECHNIQUE FOR FINDING ERRORS.

Khiste, G. P., Deshmukh, R. K. & Awate, A. P., 2018. Literature Audit of 'Digital Library': an Overview.

Laurent Candillier, K. J. F. F. F. M., 2009. State of the Art Recommender Systems. *Research Gate.*

Li, S., Kawale, J. & Fu, Y., 2015. *Deep Collaborative Filtering via Marginalized DenoisingAuto-encoder.* s.l.:s.n.

Li, S., Kawale, J. & Fu, Y., 2015. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder. pp. 812, 814.

Liu, J. & Wu, C., 2017. Deep Learning Based Recommendation: A Survey. Volume 424, p. 451.

Mishra, N. et al., 2017. Solving Sparsity Problem in Rating-BasedMovie Recommendation System. *Computational Intelligencein Data Mining,* p. 113.

Moh. Irfan, A. D. C. F. H. R., 2014. Sistem Rekomendasi: Buku Online dengan Metode Collaborative Filtering. *Jurnal Teknologi Technoscienta,* pp. 77-78.

Nahak, B. & Padhi, S., 2019. The Role of Smart Library and Smart Librarian for E- Library Services.

Rahmawati, S., Nurjanah, D. & Rismala, R., 2018. Analisis dan Implementasi Pendekatan Hybrid untuk Sistem Rekomendasi dengan Metode Knowledge Based Recommender System dan Collaborative Filtering. *Ind. Journal on Computing,* 3(2), p. 12.

Rao, V., V, R. K. & Padmanabhan, V., 2017. Divide and Transfer: Understanding Latent Factors for Recommendation Tasks. p. 1.

Rikowski, R., 2012. *Library Classification Trends in the 21st Century.* s.l.:s.n.

Saeed, M. & Mansoori, E. G., 2017. A NOVEL FUZZY-BASED SIMILARITY MEASURE FOR COLLABORATIVE FILTERING TO ALLEVIATE THE SPARSITY PROBLEM. *Journal of Fuzzy Systems,* Volume 14, p. 2.

Satija, M., 2008. *Universal Decimal Classification: Past and Present.* s.l.:s.n.

Satyaputra, M. &. E. M. A. S. K., 2016. *Lets Build Your Android Apps with Android Studio.* s.l.:Elex Media Komputindo.

Shi, Q., 2010. An Empirical Study Of Thinking Aloud Usability Testing From a Cultural Perspective.

Steven Hendrawan, e. a., 2020. Perancangan Sistem Informasi Permohonan Perizinan Penelitian dengan Metode Agile dan Framework Laravel Berbasis W. *Journal of Information Systems and Informatics.*

Suharyanto, 2012. *Dewey Decimal Classification Edisi Ke-23 : Perubahan dan Perluasan Notasi Tentang Indonesia.* s.l.:s.n.

Suresha, G. & Narayanaswamy, B., 2016. *The Scheme of Library Classifications: Concerning the Structural changes of 23rd Dewey Decimal Classification (DDC).* s.l.:s.n.

Wenige, L. & Ruhland, J., 2018. Retrieval by recommendation: using LOD technologies to improve. *digital library search,* p. 2.

Yudhanto, Y. a. A. W., 2018. *Mudah Membuat dan Berbisnis Aplikasi Android dengan Android Studio.* s.l.:Elex Media Komputindo.

Yudhanto, Y. a. A. W., 2018. *Mudah Membuat dan Berbisnis Aplikasi Android dengan Android Studio.* s.l.:Elex Media Komputindo.

Zhang, J.-D. & Chow, C.-Y., 2016. Enabling Kernel-based Attribute-aware Matrix Factorization for Rating Prediction. p. 1.

Zhang, M. & Yang, Z., 2019. GACOforRec: Session-Based Graph Convolutional Neural Networks Recommendation Model. p. 114077.

Zhang, Q. et al., 2018. CoupledCF: Learning Explicit and Implicit User-item Couplingsin Recommendation for Deep Collaborative Filtering. *International Joint Conferences on Artificial Intelligence.*

Zhang, S., 2019. *Deep Learning based Recommender System: A Survey and New Perspectives.*

Zulkarnain, T. D. A. d. A., 2013. SUGGESTIONS FRIENDS ENGINE BERBASIS HYBRID RECOMMENDER SYSTEM UNTUK MENDAPATKAN REKOMENDASI TEMAN TERBAIK PADA WEB JEJARING SOSIAL. *Jurnal Ilmiah Teknologi dan Informasi ASIA.*