# 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 PERPUSTAKAN 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?
   1. **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*.
   1. **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.
   1. **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 pengguna 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.

4). Melakukan verifikasi member terdaftar.

5). 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.

* 1. **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 Kotlin, API algoritma dengan bahasa Phython 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**

**2.1 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).

Perpustakaan digital tidak hanya terbatas pada koleksi *item* elektronik dalam bentuk cetak saja, ruang lingkup koleksinya sampai pada artefak digital yang tidak bisa digantikan dalam bentuk tercetak. Koleksi menekankan pada isi informasi, jenisnya dari dokumen tradisional sampai hasil penelusuran. Perpustakaan digital melayani mesin, manajer informasi, dan pemakai informasi. Semuanya ini demi mendukung manajemen koleksi, menyimpan, pelayanan bantuan penelusuran informasi.

**2.2 Sistem Rekomendasi**

Sistem rekomendasi merupakan fitur yang berfungsi untuk membantu pengguna untuk memberi saran kepada *user* 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>)

**2.3 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 kecendrungan pengguna, seperti kecendrungan *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.

**2.4 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).

\*apa perlu kita buat alur cara kerja algoritma pake diagram flow ? kalau ga mau ribet ambil di jurnal aja sama matrix factori….

**2.5 Matrix Factorization**

*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 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.

Sistem mempelajari *latent factors* dengan meminimalkan *objective function* sebagai berikut :



Gambar 2. 2 Sistem mempelajari *latent factors*

Di mana *l (R, U, V)* adalah fungsi untuk memprediksi *rating* menggunakan *latent factor* U dan V dan dua istilah terakhir adalah regularisasi yang digunakan untuk menghindari *overfitting*. *||· ||*F menunjukkan *Frobenius norm* (Li, et al., 2015).

**2.6 Android**

*Android* adalah sebuah sistem operasi untuk *smartphone* dan *tablet*. Sistem operasi dapat diilustrasikan sebagai 'jembatan' antara perantara (*device*) dan penggunanya, sehingga pengguna dapat berinteraksi dengan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada *device*. Di dunia personal komputer, sistem operasi yang banyak dipakai adalah *Windows*, *Mac*, dan *Linux* (Satyaputra, 2016)

Memilih sistem *Android* untuk perangkat *mobile* merupakan keputusan yang tepat karena diabawag *Google* pasti dikembangkan terus menerus dan juga menjadi *developer* yang bisa bekerja bebas membuat aplikasi berbasis *Android*. Kita juga bisa menjual produk, jasa, sebagai solusi perasalahan untuk penggunaanya, Android juga menyediakan arsitektur development yang banyak, kita tidak perlu repot menghafal komponen arsitektur dari Android, cukup mengetahui apa yang tersedia dalam sistem yang ingin dibangun dalam aplikasi kita (Yudhanto, 2018)  
Berikut cara kerja dari sistem operasi *Android* (Yudhanto, 2018)

1. Aplikasi: Aplikasi berada pada tingkat ini bersama dengan aplikasi sistem inti untuk email, perpesanan SMS, kalender, penjelajahan Internet, atau kontak.
2. *Java API Framework*: Semua fitur Android tersedia untuk developer melalui antarmuka pemograman aplikasi.
   1. Sistem tampilan digunakan untuk membangun *UI* aplikasi, termasuk daftar, tombol, dan menu.
   2. Pengelola referensi digunakan untuk mengakses sumber daya *non-code* seperti *string*, grafik, dan *file layout*.
   3. Pengelola notifikasi digunakan untuk menampilkan peringatan khusus di bilah status.
   4. Pengelola aktivitas yang mengelola daur hidup aplikasi.
   5. Penyedia materi yang memungkinkan aplikasi untuk mengakses data dari aplikasi lain.
   6. Semua *API* kerangka kerja yang digunakan aplikasi sistem *Android*.
3. *Library* dan *Android runtime*: Setiap aplikasi berjalan dalam prosesnya sendiri dan dengan *instance* *Android Runtime* sendiri, yang memungkinkan beberapa mesin sekaligus *virtual* pada perangkat bermemori rendah. *Android* juga menyertakan rangkaian *library* waktu proses inti yang menyediakan sebagian besar fungsionalitas bahasa pemrograman *Java*, termasuk beberapa fitur bahasa *Java 8* yang digunakan *framework Java API*. Banyak layanan dan komponen sistem *Android* inti dibangun dari kode asli yang memerlukan pustaka asli yang ditulis dalam *C* dan *C++.* Pustaka asli tersebut tersedia untuk aplikasi melalui kerangka kerja *Java API*.
4. *Hardware Abstraction Layer* (HAL): Lapisan ini menyediakan antarmuka standar yang menunjukkan kemampuan perangkat keras di perangkat ke kerangka kerja *Java API* yang lebih tinggi. *HAL* terdiri atas beberapa modul pustaka, masing-masing mengimplementasikan antarmuka untuk komponen perangkat keras tertentu, seperti modul kamera atau *bluetooth*.
5. *Kernel Linux*: Fondasi *platform* *Android* adalah *kernel Linux*. Lapisan di atas mengandalkan *kernel Linux* untuk fungsionalitas pokok seperti *threading* dan manajemen memori tingkat rendah. Menggunakan *kernel Linux* memungkinkan *Android* memanfaatkan fitur keamanan utama dan memungkinkan produsen perangkat mengembangkan *driver* perangkat keras untuk *kernel* yang cukup dikenal.

**2.7 Website**

*Website* website adalah sekumpulan data digital seperti teks, gambar, suara, video, yang diletakkan dalam *web server* yang umumnya diakses melalui internet (Steven Hendrawan, 2020) sebuah website terdiri dari bebapa halaman *web* yang saling berhubungan dengan web lainnya yang disebut *hyperlink*, sedangkan teks yang dijadikan media penghubung disebut *hypertext.* Domain merupakan nama unik yang dimiliki dalam sebuah institusi sehingga bisa diakses melalui internet, misalnya yahoo.com, google.com, friendster.com, dll. Untuk bisa istilah lain yang sehubungan dengan *website* adalah *homepage*. *Homepage* adalah halaman awal dari domain, misalnya, membuka *website* www.yahoo.com, halaman pertama yang muncul disebut dengan *homepage*, jika mengklik menu-menu yang ada dan meloncat ke lokasi lainnya disebut dengan *web page,* sedangkan keseluruhan isi atau konten domain disebut *website*.   
**Jenis – Jenis *Website*:**

Dalam pengelompokan jenis *web*, lebih diarahkan pada:

* + - 1. Jenis-jenis *web* berdasarkan sifatnya adalah:

1. *Website* dinamis, merupakan sebuah *website* yang menyediakan content atau isi yang selalu berubah-ubah setiap saat. Misalnya *website* berita, seperti, www.republika.co.id, [www.google.com](http://www.google.com).
2. *Website* statis, merupakan *website* yang contentnya sangat jarang diubah. Misalnya *web* profil organisasi, seperti, <http://kaskus.co.id/>
3. Ditinjau dari segi bahasa pemrograman, *website* terbagi atas:
   1. Server side, merupakan *website* yang menggunakan bahasa pemrograman yang tergantung kepada tersedianya *server*. Seperti, PHP, ASP dan sebagainya. Jika tidak ada server, *website* yang dibangun menggunakan bahasa pemograman diatas tidak dapat berfungsi sebagaimana mestinya.
   2. *Client side*, adalah *website* yang tidak membutuhkan *server* dalam menjalankannya, cukup diakses melalui browser saja. Misalnya, html.
4. Berdasarkan tujuannya*, website* dibagi atas:
   1. *Personal web*, *website* yang berisi informasi pribadi seseorang.
   2. *Corporate web*, *website* yang dimiliki oleh sebuah perusahaan.
   3. *Portal web*, *website* yang mempunyai banyak layanan, mulai dari layanan berita, email, dan jasa-jasa lainnya.
   4. *Forum web*, sebuah *web* yang bertujuan sebagai media diskusi.

**2.8 Pengamanan Data ?**

**2.7 Pengujian Perangkat Lunak ?**

**2.8 Dataset ?**

# DAFTAR PUSTAKA

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.*

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 ).*

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.

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

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.

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.

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.

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

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.*

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

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.*