REKOMENDASI FILM BERDASARKAN SINOPSIS MENGGUNAKAN METODE *WORD2VEC*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Alimah Nur Laili
NIM: 155150201111136

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

REKOMENDASI FILM BERDASARKAN SINOPSIS MENGGUNAKAN METODE *WORD2VEC*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh:
Alimah Nur Laili
NIM: 155150201111136

Skripsi ini telah diuji dan dinyatakan lulus pada
09 Juli 2019
Telah diperiksa dan disetujui oleh:

Pembimbing I	Pembimbing II
Putra Pandu Adikara, S.Kom., M.Kom.	Sigit Adinugroho, S.Kom., M.Sc.
NIP: 19850725 200812 1 002	NIK: 201607 880701 1 000

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 21 Juni 2019

Alimah Nur Laili

NIM: 155150201111136

PRAKATA

Puji syukur atas kehadirat Allah SWT, karena berkat rahmat dan hidayah-NYA penulis dapat menyelesaikan tanggung jawab penelitian skripsi yang berjudul "Rekomendasi Film Berdasarkan Sinopsis Menggunakan Metode Word2Vec".

Dalam proses menyelesaikan penelitian ini, penulis mendapat banyak bantuan, dukungan serta doa dari beberapa pihak, untuk itu dengan rasa hormat penulis ingin menyampaikan ucapan terima kasih kepada:

1. Bapak Putra Pandu Adikara, S.Kom., M.Kom dan Bapak Sigit Adinugroho, S.Kom., M.Sc selaku pembimbing 1 serta pembimbing 2 skripsi dari penulis karena telah banyak membimbing, membantu serta memotivasi untuk dapat menyelesaikan skripsi ini dengan baik.
2. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Agus Wahyu Widodo, S.T., M.Sc. selaku Ketua Program Studi Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Aswadi dan Ibu Wuryan Sari selaku orang tua penulis, serta adik-adik penulis Abiyyu Helmi, Zayyim Arief dan Salsabila Nur Addina yang selalu memberikan bantuan, motivasi dan doa selama penulis menyelesaikan skripsi.
6. Muhammad Heryan Chaniago, Nestiti Praneswari, Shafira Dwita, Karina Natasha, Astya Dhia, Antania Aswarani, Tiara Ramadhanty, Rifni Ayu, Masofia Rizky, Dewi Cahya, Intan Permata, Ristievanny Tikupadang dan Melati Ayuning yang selalu memberikan bantuan, semangat, dan doa selama penyelesaian skripsi.
7. Seluruh badan pengurus Himpunan Mahasiswa Informatika (HMIF) yang selalu memberikan pengertian, doa, serta dukungan selama penyelesaikan skripsi.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih terdapat banyak kekurangan dan jauh dari kata sempurna, maka dari itu penulis mengharapkan kritik serta saran yang dapat membangun. Sebagai penutup, penulis berharap skripsi ini bermanfaat untuk pengembangan penelitian selanjutnya maupun semua orang.

Malang, 21 Juni 2019

Penulis
Email: alimahnrl@gmail.com

ABSTRAK

Alimah Nur Laili, Rekomendasi Film Berdasarkan Sinopsis Menggunakan Metode *Word2Vec*

Pembimbing: Putra Pandu Adikara, S.Kom., M.Kom. dan Sigit Adinugroho, S.Kom., M.Sc.

Jumlah film yang tersebar di seluruh dunia mengalami peningkatan produksi tiap tahunnya, hal tersebut menunjukkan bahwa minat masyarakat terhadap dunia perfilman semakin tinggi. Melakukan sebuah pencarian data dengan parameter tertentu di internet, tentu sulit untuk mendapatkan hasil yang sesuai dengan keinginan karena banyaknya data yang tersedia namun terbatasnya alat yang memadai. Penyaringan informasi data yang berlebih dapat dilakukan menggunakan proses rekomendasi. Terdapat beberapa tahap pada proses rekomendasi film yaitu tahap *Pre-processing* untuk memproses dokumen sinopsis film, metode TF-IDF untuk mendapatkan kata dengan bobot tertinggi sebanyak jumlah yang ditentukan bedasarkan hasil kecocokan *query* didalam dokumen, *Word2Vec* sebagai metode untuk mendapatkan *Query Expansion* dari hasil kata teratas yang diambil pada proses TF-IDF dan *Cosine Similarity* untuk mendapatkan nilai kemiripan dokumen dengan *query*. Metode *Word2Vec* berfungsi mencari nilai kedekatan antar kata satu sama lain untuk mendapatkan kata yang akan ditambahkan kedalam *query* awal. Data latih berupa judul dengan sinopsis film berbahasa Inggris berjumlah 150, proses evaluasi mengambil 30 data judul dan sinopsis dari data latih berdasarkan film yang dipilih oleh penguji. Nilai rata-rata *Precision@*10 tertinggi yang diperoleh sebesar 0,47 dan nilai *Mean Average Precision* (MAP) tertinggi diperoleh sebesar 0,709603374.

Kata Kunci: *rekomendasi, film, query expansion, word2vec, TF-IDF, cosine similarity*

ABSTRACT

Alimah Nur Laili, Rekomendasi Film Berdasarkan Sinopsis Menggunakan Metode *Word2Vec*

Advisor: **Putra Pandu Adikara, S.Kom., M.Kom.** ***and*** **Sigit Adinugroho, S.Kom., M.Sc.**

*The number of movie production have increased each year. This shows that the society interest in the film industry is getting higher. It's difficult to get the appropriate result of what desired by searching for data with certain parameters on the internet because of the large amount of data exists but there is limited adequate tools. The screening of the excess data can be done using recommendation process. There are several stages in movie recommendation process. Those are Pre-processing to process film synopsis documents, TF-IDF method to obtain the highest value as much as the amount determined based on the query result on the document. Word2vec as a method to get the query expansion from the top word result that taken from TF-IDF process and Cosine Similarity is used to get the similarity between document and query. The Word2Vec method plays role to find the proximity value between words to one another in order to get the words that will be added to the initial query. The training data are 150 movies title with English synopsis. The evaluation process took 30 data of movie title and synopsis from the training data based on the movies selected by the examiners. The highest Precision@*10 *value is 0,47 and the highest Mean Average Precision (MAP) value is 0.709603374.*

Keywords: *recommendation, movie, query expansion, word2Vec, TF-IDF, cosine similarity*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar belakang

Perkembangan teknologi pada era globalisasi khususnya bidang teknologi informasi berkembang dengan pesat. Film merupakan gabungan dari perkembangan teknologi audio visual yang dijadikan sebagai salah satu media hiburan maupun penyampaian informasi dan selalu populer di kalangan masyarakat. Menonton film favorit dapat berfungsi untuk merevitalisasi diri sehingga dapat mengembalikan kepercayaan diri untuk dapat beraktivitas lebih baik (Agrawal dan Jain, 2017). Jumlah film yang tersebar di seluruh dunia terus mengalami peningkatan setiap tahunnya. Judul film yang telah dikeluarkan oleh industri perfilman pada halaman web *The Movie Database* (TMDb) mulai tahun 1900 sampai tahun 2019 terdapat sebanyak 14.971 (*The Movie Database*, 2019). Semakin banyak produksi film yang dihasilkan menunjukkan bahwa semakin tinggi minat masyarakat terhadap dunia perfilman.

Minat masyarakat terhadap film diiringi dengan perkembangan teknologi yang semakin canggih serta meningkatnya kebutuhan dalam dunia digital, menghasilkan banyak proses bisnis yang melibatkan dunia perfilman serta dunia digital antara lain Iflix, Netflix, HOOQ, Youtube dan lain-lain. Setiap orang memiliki kecenderungan menonton sebuah film berdasarkan alir cerita dan pembawaan sinematiknya. Sinopsis merupakan salah satu parameter yang dapat membantu seseorang dalam mengetahui garis besar alir cerita serta pembawaan sinematik dari sebuah film. Melakukan pencarian terhadap film dengan sinopsis yang memiliki kemiripan secara manual, akan membutuhkan waktu yang lama karena banyaknya data yang tersedia pada jejaring sosial di internet serta terbatasnya alat yang memadai (Agrawal dan Jain, 2017). Rekomendasi merupakan alat penyaringan informasi dari jejaring sosial yang dapat digunakan untuk mengatasi data yang berlebih (Roy dan Kundu, 2013). Proses rekomendasi banyak digunakan pada penyedia layanan *streaming* digital sebagai fitur yang digunakan untuk meningkatkan kepuasan pelanggan terhadap produk/jasa yang ditawarkan. Rekomendasi akan menyaring informasi yang tidak relevan dan menampilkan informasi yang relevan sesuai keinginan pengguna.

Terdapat beberapa peneliti yang pernah meneliti proses rekomendasi yaitu penelitian pertama (Saksono, Sari dan Dewi, 2018) mengenai rekomendasi lokasi wisata kuliner menggunakan metode *K-Means Clustering* yang berfungsi untuk membagi lokasi wisata bedasarkan jarak dan *Simple Additive Weighting* yang berfungsi untuk mengurutkan jarak lokasi sesuai keinginan *user*. Hasil pengujian dengan menggunakan nilai akurasi mendapatkan presentase sebesar 63,33%, 40% dan 46,67% pada kategori sangat dekat, dekat dan sedang. Nilai akurasi dihasilkan dari perhitungan antara hasil yang direkomendasikan dengan hasil yang dipilih oleh 30 responden. Penelitian kedua mengenai rekomendasi bahan makanan bagi penderita penyakit jantung (Siahaan, Cholissodin dan Fauzi, 2017), hasil rekomendasi berupa bahan makanan untuk 5 kali makan dalam 1 hari dan

memenuhi kebutuhan gizi tidak lebih dari 10% untuk penderita penyakit jantung. Penelitian tersebut menggunakan algoritme genetika dengan metode *crossover* berupa *extended intermediate crossover*, metode mutasi berupa *random mutation* dan metode seleksi menggunakan *elitism selection*, menghasilkan nilai parameter paling optimal pada nilai *fitness* sebesar 103,7 dengan jumlah populasi 280, nilai Cr sebesar 0,5, nilai Mr sebesar 0,5, rata-rata nilai *fitness* sebesar 103,3 dan rata-rata nilai *fitness* pada generasi 100 sebesar 111,2. Penelitian ketiga (Ayu, Putri dan Supianto, 2019) mengenai *query expansion* pada LINE TODAY, menggunakan teknik *pre-processing* sebagai proses awal, lalu teknik pembobotan term dan *cosine similarity*. Dalam menghasilkan *query* tambahan, metode yang digunakan adalah metode turunan dari *Rocchio relevance feedback* yaitu *extended rocchio relevance feedback*. Hasil dari proses pengujian, mendapatkan nilai *Precision* sebesar 0,53308, nilai *Recall* sebesar 0,81708, nilai *F-Measure* sebesar 0,59553 dan nilai akurasi sebesar 0,9574.

Berdasarkan penelitian sebelumnya, penulis akan melakukan penelitian mengenai rekomendasi film bedasarkan sinopsis menggunakan tahap *pre-processing* untuk memproses dokumen sinopsis film, metode TF-IDF untuk mendapatkan kata dengan bobot tertinggi sebanyak jumlah yang ditentukan bedasarkan hasil kecocokan *query* didalam dokumen, *Word2Vec* sebagai metode untuk mendapatkan *query expansion* dari hasil kata teratas yang diambil pada proses TF-IDF dan *cosine similarity* untuk mendapatkan nilai kemiripan dokumen dengan *query*. Terdapat beberapa penelitian sebelumnya yang pernah dilakukan menggunakan metode *Word2Vec*, TF-IDF dan *Cosine Similarity*.

Penelitian pertama mengenai *Word2Vec* dilakukan oleh (Mikolov et al. 2013b) menghasilkan sebuah model dari representasi kata dan frasa terdistribusi serta komposisionalitasnya yaitu model *Skip-gram*. Model *Skip-gram* dinilai efisien secara komputasi dalam mempelajari representasi vektor kata berdimensi tinggi serta menangkap kata yang memiliki hubungan kata secara semantik dan sintaksis secara tepat, model ini juga dapat digunakan untuk melakukan penambahan vektor kata dan menghasilkan hasil kata yang lebih bermakna. Penelitan selanjutnya (Laxmi, Indriati dan Fauzi, 2019) mengenai *query expansion* pada sistem temu kembali informasi berbahasa indonesia dengan metode pembobotan TF-IDF dan algoritme *cosine similarity* berbasis *WordNet* menghasilkan perbandingan antara nilai *precision*@k menggunakan *query expansion* denggan tidak menggunakan *query expansion*. Hasil menunjukkan bahwa nilai rata-rata *precision*@20 dari 10 *query* dengan menggunakan *query expansion* sebesar 70%, lebih besar dibandingkan tidak menggunakan *query expansion* 52%.

Bedasarkan uraian tersebut, penulis memutuskan untuk membuat suatu penelitian mengenai Rekomendasi Film Berdasarkan Sinopsis dengan metode TF-IDF dan *Cosine Similarity* serta penambahan *Query Expansion* menggunakan metode *Word2Vec*. Hasil dari penelitian ini diharapkan dapat menghasilkan hasil rekomendasi yang sesuai dengan keinginan pengguna.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya, rumusan masalah dari penelitian ini yaitu:

1. Bagaimana merancang algoritme *Word2Vec* untuk proses *Query Expansion*?
2. Bagaimana hasil nilai rata-rata *Precision dan Mean Average Precision* pada proses rekomendasi film?
3. Bagaimana hasil nilai rata-rata *Precision*, *dan Mean Average Precision* pada proses rekomendasi film saat menggunakan proses *Query Expansion* dengan tidak menggunakan proses *Query Expansion*?

1.3 Tujuan

Berdasarkan pada rumusan masalah yang telah disebutkan, tujuan yang ingin dicapai dari penelitian ini adalah:

1. Menerapkan metode *Word2Vec* untuk mendapatkan *Query Expansion* untuk proses rekomendasi.
2. Mengetahui nilai rata-rata *Precision dan Mean Average Precision* yang dihasilkan dalam proses perekomendasian film.
3. Mengetahui nilai rata-rata *Precision dan Mean Average Precision* yang dihasilkan dalam proses perekomendasian film pada saat menggunakan proses *Query Expansion* dengan tidak menggunakan proses *Query Expansion.*

1.4 Manfaat

Manfaat yang dihasilkan dari penelitian ini yaitu:

1. Membantu masyarakat untuk menemukan rekomendasi film yang sesuai dengan keinginan.
2. Membuktikan bahwa metode *Word2Vec* dapat menghasilkan *Query Expansion* yang akurat untuk proses rekomendasi.

1.5 Batasan masalah

Batasan-batasan yang digunakan untuk menghidari perluasan permasalahan yaitu:

1. Sumber data didapatkan dari halaman web www.moviedb.com
2. Data yang dipakai menggunakan sinopsis dari film-film berbahasa Inggris yang ada pada halaman web.
3. Teknik yang digunakan pada *Word2Vec* menggunakan model *Skip-Gram*.

1.6 Sistematika pembahasan

Bagian ini berisi struktur penelitian ini mulai Bab Pendahuluan sampai Bab Penutup dan deskripsi singkat dari masing-masing bab. Diharapkan bagian ini

dapat membantu pembaca dalam memahami sistematika pembahasan isi dalam tugas akhir ini.

BAB I : PENDAHULUAN

Berisi uraian mengenai latar belakang, perumusan masalah, tujuan dari penelitian, manfaat penelitian, batasan masalah dan sistematika pembahasan.

BAB II : LANDASAN KEPUSTAKAAN

Berisi uraian tentang dasar teori penelitian sebelumnya yang berhubungan dengan penelitian rekomendasi film serta teori pendukung mengenai film, *Text Mining*, rekomendasi, *Query Expansion*, metode TF-IDF, metode *Word2Vec*, metode *Cosine Similarity* dan metode evaluasi.

BAB III : METODOLOGI PENELITIAN

Berisi penjelasan tentang metode yang digunakan serta langkah-langkah yang digunakan dalam penelitian, terdiri dari tipe penelitian, strategi dan rancangan penelitian, lokasi penelitian, teknik pengumpulan data, teknik analisis data, peralatan pendukung serta implementasi algoritme penelitian.

BAB IV : PERANCANGAN DAN IMPLEMENTASI

Berisi tentang proses manualisasi perhitungan metode, perancangan dari algoritme dan evaluasi yang digunakan dan proses implementasi algoritme.

BAB V : PENGUJIAN DAN ANALISIS

Berisi penjelasan proses mengenai pengujian dari hasil implementasi serta analisis hasil dari pengujian terkait metode yang digunakan.

BAB VI : PENUTUP

Berisi kesimpulan dari hasil penelitian yang telah dibuat serta saran yang diberikan untuk penelitian selanjutnya.

BAB 2
LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

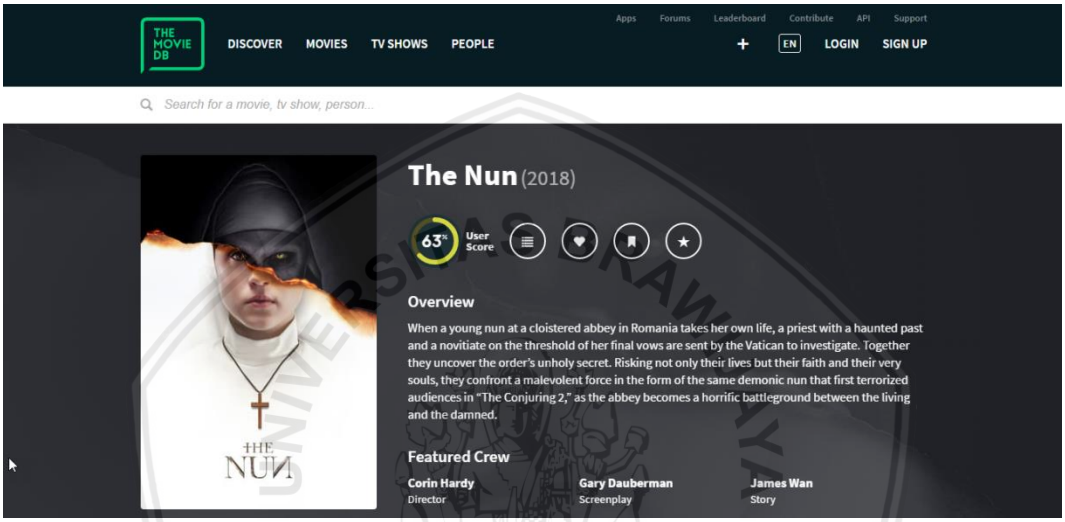
Kajian pustaka berisi penjelasan mengenai penelitian yang pernah dilakukan sebelumnya, terdapat beberapa penelitian yang didapat berdasarkan referensi melalui skripsi atau jurnal. Penelitian pertama yaitu mengenai metode *Word2Vec* yang dilakukan oleh (Mikolov et al. 2013b) menghasilkan sebuah model dari representasi kata dan frasa terdistribusi serta komposisionalitasnya yaitu model *Skip-gram*. Model *Skip-gram* dinilai efisien secara komputasi dalam mempelajari representasi vektor kata berdimensi tinggi, memperdulikan urutan kata pada sebuah kalimat sehingga dapat menangkap kata yang memiliki hubungan secara semantik dan sintaksis secara tepat, model ini juga dapat digunakan untuk melakukan penambahan vektor kata dan menghasilkan hasil kata yang lebih bervariasi dan bermakna. Hasil pengujian yang dapat memengaruhi kinerja yaitu pemilihan arsitektur model yang digunakan, ukuran dari vektor, *subsampling rate*, dan ukuran dari *training window*.

Penelitian kedua (Laxmi, Indriati, dan Fauzi, 2019) mengenai *query expansion* pada sistem temu kembali informasi berbahasa indonesia dengan metode pembobotan TF-IDF dan algoritme *cosine similarity* berbasis *WordNet*. Hasil pengujian menghasilkan perbandingan antara nilai *precision*@k menggunakan *query expansion* denggan tidak menggunakan *query expansion*. Hasil menunjukkan bahwa nilai rata-rata *precision*@20 dari 10 *query* dengan menggunakan *query expansion* sebesar 70%, lebih besar dibandingkan tidak menggunakan *query expansion* 52%. Hal tersebut menunjukkan bahwa menambahkan proses *query expansion* dapat membuat hasil kerja sistem lebih baik.

Penelitian ketiga (Rattinger, Le Goff, dan Guetl, 2018) mengenai *Local Word Embedding for Query Expansion* berdasarkan *Co-Authorship* dan *Citations* menyebutkan bahwa percobaan menggunakan dua kumpulan data yaitu ACL (*Anthology Collection*) berupa dokumen publikasi ilmiah dibidang linguistik komputasi sebanyak 33.922 artikel serta dokumen berbahasa inggris dari CLEF-IP 2011 *Collection* berupa 400.000 paten dan 1.350 topik. Peneliti menggunakan algoritme *Skip-Gram* dari *Word2Vec*, hasil menunjukkan bahwa, *Word Embedding* yang melalui tahap *local training* memberikan augmentasi yang berhaga serta meningkatkan kinerja dari *retrieval performance*. Pengujian dilakukan pada 4 kondisi yaitu pengujian tanpa metode *Query Expansion*, pengujian *Query Expansion Global* dari pelatihan *dataset* Wikipedia bahasa inggris, pengujian *Query Expansion local* dari pelatihan secara lokal dan *Query Expansion local training* extension dengan perpanjangan dokumen. Nilai MAP (*Mean Average Precision*) dan nilai *Precision*@10 yang dihasilkan untuk dokumen yang diambil melalui *Query Expansion Local Training Extension* menghasilkan nilai terbesar yaitu 0,1713 dan 0,1822, sedangkan nilai *precision*@5 terbesar dihasilkan pada pengujian *Query Expansion local* sebesar 0,2347.

Penelitian terakhir (Nasution, Bijaksana dan Al Faraby, 2017) yaitu mencari kemiripan antar ayat dalam Al-Quran menggunakan *word alignment*. Peneliti menggunakan *database* parafrase sebagai tambahan data, merepresentasikan proses penyejajaran ayat menjadi vektor kata dengan *word2vec* dan mengukur nilai kemiripan menggunakan metode *cosine similarity*. Dalam menghitung nilai prediksi berdasarkan *word alignment* serta *word2vec*, metode yang digunakan yaitu *Support Vector Regression* (SVR). Hasil penelitian menunjukkan nilai *pearson correlation* yang didapat sebesar 0,81221.

2.2 Film



Gambar 2.1 Salah satu sinopsis film The Nun.

Menurut Kamus Besar Bahasa Indonesia, arti dari film adalah gambar hidup (*KBBI-Online*, 2019). Film merupakan sebuah karya hasil berfikir kreatif dari seorang sineas yang didalamnya mengandung tingkah laku manusia, nilai kehidupan, norma-norma kehidupan, pandangan hidup, serta kecanggihan teknologi (Trianton, 2013). Film merupakan suatu media hiburan yang banyak diminati oleh masyarakat, dengan memadukan audio serta visual yang baik dapat memberikan sebuah kekuatan komunikasi sehingga pesan serta misi yang terkandung dalam sebuah film dapat tersampaikan dengan baik (Wahyudi, 2018). Masing-masing film dibedakan berdasarkan genre seperti drama, komedi, thriller, horor, dan lain-lain. Seiring dengan meningkatnya minat masyarakat terhadap film, serta kebutuhan akan dunia digital menghasilkan industri bisnis yang melibatkan film memiliki harga jual dan konsumen tetap seperti Netfilx, Iflix, HOOQ dan lain-lain.

Dunia digital mendasari banyaknya *website* maupun aplikasi penyedia informasi yang menyediakan segala informasi mengenai suatu film seperti judul, pemain yang terlibat, sutradara, genre, sinopsis dan lain-lain. Sinopsis merupakan suatu ringkasan atau garis besar naskah yang menggambarkan isi dari suatu film, sinopsis sebuah film dapat membantu penonton untuk menentukan pilihan sebuah film yang akan ditonton. Judul film yang telah dikeluarkan oleh industri

perfilman pada halaman web *The Movie Database* (TMDb) mulai tahun 1900 sampai tahun 2019 terdapat sebanyak 14.971 (*The Movie Database*, 2019).

2.3 *Text Mining*

Text Mining merupakan salah satu pengaplikasian data *mining* yang khusus untuk mengolah data dalam bentuk teks yang tidak terstruktur. *Text Mining* memiliki fungsi yaitu dapat mengubah struktur kata menjadi terstruktur dari struktur kata yang tidak struktur, proses yang terdapat pada *Text Mining* yaitu *pre-processing*, *pattern discovery*, *text transformation*, *feature selection* dan *evaluation* (Berry dan Kogan, 2010). *Text mining* dapat dijadikan solusi pada krisis informasi yang berlebihan dengan melakukan sebuah penggabungan antara teknik *Data Mining*, *Information Retrieval*, *Natural Language Processing*, *Machine Learning*, dan *Knowledge Management* (Susilowati, Sabariah dan Gozali, 2015). Hasil dari *Text Mining* nantinya dapat digunakan untuk merekomendasikan sebuah film.

2.3.1 *Text Pre-processing*

Dalam *text mining*, tujuan dilakukannya proses text *pre-processing* yaitu untuk dapat menghasilkan sebuah data yang berkualitas agar lebih mudah dieksekusi, *pre-processing* dilakukan dengan cara menghilangkan kata yang tidak mengandung arti atau kata yang tidak penting, terdapat beberapa tahap yang ada dalam *pre-processing* yaitu *tokenization*, *case folding*, *filtering/ stopword removal* dan *stemming* (Berry dan Kogan, 2010). Penelitian ini menggunakan library NLTK untuk proses *pre-processing* yaitu proses *tokenisasi*, *case folding* dan *stopword removal*. Penjelasan tahap *pre-processing* terdapat pada Subbab 2.4.1.1 sampai 2.4.1.3.

2.3.1.1 Tokenisasi

Tokenisasi merupakan suatu proses pemisahan kata dari sebuah kalimat menjadi potongan kata. Pada proses tokenisasi menggunakan konsep pemisahan kata berdasarkan spasi untuk melakukan pemisahan kata serta menghapus beberapa karakter tertentu yang dianggap sebagai tanda baca.
Contoh: Film harry potter pada serial ke lima.
Tokenisasi: Film // harry // potter // pada // serial // ke // lima.

2.3.1.2 *Case Folding*

Case folding digunakan untuk mengubah semua huruf yang terdapat dalam dokumen menjadi huruf kecil (*lowercase*), hal ini dilakukan agar penyamarataan makna dari setiap kata.
Contoh: Film Harry potter Pada serial ke lima.
Case folding: film // harry // potter // pada // serial // ke // lima

2.3.1.3 *Stopword Removal*

Stopword removal merupakan suatu proses pembuangan kata yang sering muncul namun memiliki arti yang tidak relevan sehingga tidak mampu mewakili sebuah dokumen, seperti kata 'yang', 'di', 'ke' dll.

Contoh: Film harry potter pada serial ke lima.

Stopword removal: film // harry // potter // serial // lima

2.4 Pembobotan kata

Tujuan dari adanya pembobotan kata agar setiap kata diberi bobot sesuai frekuensi kemunculkan kata dalam suatu dokumen. Terdapat beberapa tahapan yang ada pada proses pembobotan kata, penjelasan proses terdapat pada Subbab 2.4.1 sampai 2.4.5 (Manning, Schütze dan Raghavan, 2008).

2.4.1 *Term frequency* ($tf_{t,d}$)

Term frequency menyatakan jumlah kata/term/token yang muncul pada sebuah dokumen. Untuk menghitung bobot dari $tf_{t,d}$ dapat menggunakan Persamaan 2.1.

$$tf_{t,d} \begin{cases} 1 + log_{10}(f_{t,d}), jika\ f_{t,d} > 0 \\ 0, jika\ f_{t,d} = 0 \end{cases} \quad (2.1)$$

Keterangan:

$f_{t,d}$ = frekuensi term (*t*) pada dokumen (*d*)

2.4.2 *Document frequency* (df_t)

Document frequency menyatakan banyaknya dokumen dimana suatu kata/term/token muncul.

2.4.3 *Inverse document frequency* (idf_t)

Inverse document frequency menyatakan tingkat kepentingan suatu kata, semakin banyak munculnya suatu kata dalam sebuah dokumen dianggap tidak memiliki arti yang penting, namun pada kata yang sedikit muncul pada sebuah dokumen mempunyai nilai *inverse document frequency* yang tinggi. Untuk menghitung idf_t dapat menggunakan Persamaan 2.2.

$$idf_t = log_{10} \frac{N}{df_t} \quad (2.2)$$

Keterangan:

N = jumlah banyaknya seluruh dokumen

df_t = jumlah banyaknya dokumen yang mengandung term (*t*)

2.4.4 *Term frequency-invers document frequency* ($W_{t,d}$)

Bobot yang dihasilkan oleh *Term frequency-invers document frequency* (TF-IDF) merupakan hasil perkalian dari bobot *Term frequency* ($tf_{t,d}$) dengan *Inverse document frequency* (idf_t) melalui Persamaan 2.3.

$$W_{t,d} = tf_{t,d}\ idf_t \quad (2.3)$$

Keterangan:

$W_{t,d}$ = bobot term (*t*) terhadap dokumen (*d*)

$tf_{t,d}$ = jumlah kemunculan term (*t*) dalam dokumen (*d*)

idf_t = nilai *inverse document frequency*

2.4.5 Normalisasi

Nilai bobot yang dihasilkan dari *Term frequency-invers document frequency* dilakukan proses normalisasi data menggunakna rumus pada Persamaan 2.4.

$$W_{t,d} = \frac{W_{t,d}}{\sqrt{\sum_{t=1}^{n}(W_{t,d})^2}} \quad (2.4)$$

Keterangan:

$W_{t,d}$ = bobot term (*t*) terhadap dokumen (*d*)

n = banyaknya kata

2.5 *Query Expansion*

Query Expansion merupakan suatu teknik penambahan *query* pada *Information retrieval*, yang merepresentasikan sebuah informasi ke dalam *query* yang sesuai sehingga dapat memperkuat hasil pencarian. *Query Expansion* dapat dilakukan dengan cara menambah, memperbaiki, maupun membuang bobot tiap kata pada *query* menjadi lebih relevan agar sesuai dengan keinginan pengguna. Konsep penambahan *query* yaitu mencari kedekatan tiap kata atau sinonim dalam bentuk *unstemmed-term* (Laxmi, Indriati dan Fauzi, 2019). Menurut (Manning, Schütze dan Raghavan, 2008) *Query Expansion* dapat diartikan sebagai pemberian tanda untuk dokumen relevan dari penambahan sebuah *input* pengguna pada koleksi dokumen. Ilustrasi dari penambahan *query* adalah sebagai berikut:

Kalimat	: Akhir cerita, Iron Man dapat mengalahkan Thanos.
Query yang ingin ditambah	: Thanos.
Query tambahan	: merebut, batu, ajaib.
Hasil *Query Expansion*	: Akhir cerita, Iron Man dapat mengalahkan Thanos merebut batu ajaib.

2.6 Rekomendasi

Rekomendasi merupakan suatu bentuk pendukung seseorang dalam melakukan pengambilan keputusan. Rekomendasi akan menghasilkan hasil yang relevan serta berguna bagi pengguna. Banyak penelitian mengenai rekomendasi telah dilakukan, terdapat beberapa metode yang dapat digunakan dalam melakukan rekomendasi pada berbagai macam bidang yaitu *content-based filtering*, *collaborative filtering* dan *Hybrid* (Su dan Khoshgoftaar, 2009).

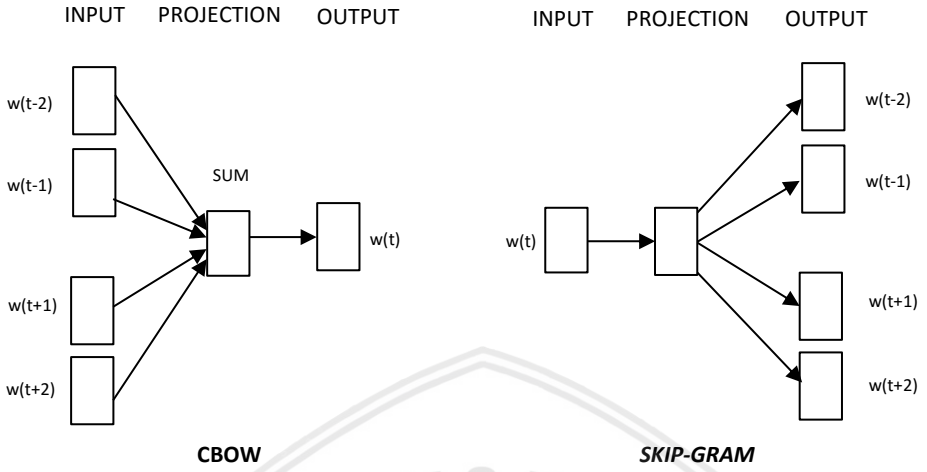
Pada *content-based filtering*, akan mendefinisikan profil suatu objek yang nantinya ditemukan sebuah rekomendasi profil sesuai dengan atribut yang diinginkan pengguna serta mencocokkan atribut yang terdapat pada masing-masing objek. Berbeda dengan *collaborative filtering* yang bergantung pada aktivitas pengguna, dan berakhibat dapat memengaruhi objek lainnya dalam pengambilan keputusan, maka dari itu pemberian rating oleh *user* sebelumnya dapat memengaruhi hasil rekomendasi. Metode *Hybrid* merupakan metode yang menggabungkan *Content Based* dan *Collaborative Filtering* untuk mendapatkan hasil rekomendasi yang lebih akurat (Tang, Hu & Liu, 2013). Penelitian ini menggunakan metode *Content Based Filtering* karena menggunakan film sebagai objek yang akan direkomendasikan sesuai dengan sinopsis yang diinginkan pengguna. Pengguna diminta untuk memasukkan film dengan judul yang diinginkan, lalu sistem akan mendeteksi sinopsis dari film tersebut, setelah terdeteksi sistem akan mencari nilai kemiripan antara sinopsis yang diinginkan pengguna dengan sinopsis film yang ada pada data latih, hasil film yang memiliki jarak terdekat berdasarkan sinopsisnya akan ditampilkan sebagai hasil rekomendasi kepada pengguna. Hasil pencarian film akan ditampilkan berupa judul film dari sinopsis yang memiliki kedekatan aatau nilai tertinggi sebanyak jumlah tertentu, hasil pencarian tersebut yang akan dijadikan rekomendasi untuk pengguna.

2.7 *Word2Vec*

Word2vec merupakan representasi dari *Word Embedding* yaitu sebuah algoritme pembelajaran mesin yang diperkenalkan oleh (Mikolov et al. 2013a) dan Google untuk mendapatkan sebuah vektor berkualitas tinggi dalam merepresentasikan fitur kata dengan panjang variable tetap (*fixed-length*) dari potongan-potongan teks seperti kalimat, paragraf atau dokumen. Algoritme *Word2Vec* mewakili setiap dokumen oleh vektor kata yang melalui tahap pelatihan untuk memprediksi kata yang ada dalam dokumen. Pada proses prediksi, vektor suatu paragraf disimpulkan dengan memperbaiki vektor kata dan melatih vektor paragraf baru hingga mencapai konvergensi. *Word2Vec* dapat mengatasi kelemahan yang ada pada model *Bag-of-words* yaitu dapat mengurangi *sparse matrix* karena *Word2Vec* memperhatikan urutan kata serta dapat mempertahankan makna semantik maupun sintaksis sebuah dokumen sehingga menghasilkan kedekatan yang lebih baik terhadap kata-kata.

Konsep kerja dari teknik *Word2Vec* menyerupai teknik dari *Neural Network* dimana terdapat dua model yang dikenalkan yaitu *Continuous Bag-of Words* (CBOW) dan *Skip-Gram* (Mikolov et al. 2013a). Perbedaan dari kedua model tersebut yaitu apabila pada CBOW, saat ingin memprediksi sebuah kata maka akan diberikan sebuah "konteks" sedangkan pada *Skip-Gram* saat inigin memprediksi sebuah konteks maka akan diberikan sebuah kata (Putra, 2018). Pada model CBOW terdapat beberapa *input* yang nantinya akan menjadi sebuah *output* yang lebih spesifik sedangkan pada *Skip-Gram*, *input* akan menjadi beberapa *output* yang memiliki similaritas. Pada penelitian ini menggunakan model *Skip-Gram* karena model ini lebih cocok untuk melakukan proses penambahan *query* karena

model *Skip-Gram* akan mencari kata konteks dari sebuah kata target yang dimasukkan sehingga akan mendapatkan kata-kata sesuai tetangga terdekat. Arsitektur model *Skip-Gram* dan CBOW digambarkan pada Gambar 2.2.



Gambar 2.2 Arsitektur model CBOW dan *Skip-Gram*.
Sumber : (Mikolov et al. 2013a)

Terdapat beberapa tahapan algoritme *Word2Vec* dengan model *Skip-Gram*, yaitu:

1. Melakukan inisialisasi *window size*, *learning rate* dan jumlah *hidden neuron*.
2. Melakukan inisialisasi vektor kata dari kata target (X_t) dan vektor kata konteks (X_c). Kata akan bernilai 1 apabila termasuk ke dalam kata target ataupun kata konteks sedangkan kata lainnya diberi nilai 0. Banyaknya kata konteks berdasarkan jumlah dari *window size*.
3. Melakukan inisialisasi nilai bobot awal matriks konteks (W) dan matriks *embedding* (W') dengan membangkitkan nilai secara acak.
4. Pada setiap *epoch*, lakukan proses 5-9.
5. Pada setiap kata yang menjadi kata target, lakukan proses 6 sampai 9.
6. Hitung *Foward Pass*
   a. Hitung nilai bobot dari *input layer* ke *hidden layer* menggunakan Persamaan 2.5.

$$h = W^T X_t \tag{2.5}$$

   Keterangan:
   h = *hidden layer*
   X_t = *input* vektor kata target
   W^T = bobot *input layer* ke *hidden layer* yang ditransposisi

   b. Hitung nilai bobot dari *hidden layer* ke *output layer* menggunakan Persamaan 2.6.

$$u_j = W'^T h \tag{2.6}$$

   Keterangan:
   u_j = *output* baris ke-*j*
   h = *hidden layer*
   W'^T = bobot *hidden layer* ke *output layer* yang ditransposisi

7. Hitung *Softmax Output* menggunakan Persamaan 2.7.

$$y_j = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})} \quad (2.7)$$

Keterangan:

y_j = nilai *softmax output* baris ke-*j*
u_j = nilai *output* baris ke-*j*
$u_j{}'$ = nilai *output* seluruh baris
V = jumlah dari kata unik

8. Hitung Nilai *error* menggunakan Persamaan 2.8.

$$e = \sum_1^c (y_j - X_c) \quad (2.8)$$

Keterangan:

e = nilai *error*
x_c = nilai vektor kata konteks
y_j = nilai *softmax output* baris ke-*j*

9. Hitung *Backpropagation*
   a. Hitung nilai bobot dari *output layer* ke *hidden layer* menggunakan Persamaan 2.9.

$$dl'_{dw} = h \ e^T \quad (2.9)$$

Keterangan:

dl'_{dw} = bobot *output layer* ke *hidden layer*
h = nilai *hidden layer* dari hasil *foward pass*
e^T = nilai *error* yang ditransposisi

   b. Hitung nilai bobot dari *hidden layer* ke *input layer* menggunakan Persamaan 2.10.

$$dl_{dw} = X_c \ (W' \ e)^T \quad (2.10)$$

Keterangan:

dl_dw = bobot *input layer* ke *hidden layer*
X_c = *input* vektor kata *embedding*
e = nilai *error*
W' = bobot *hidden layer* ke *output layer*

10. Hitung pemutakhiran bobot
    a. Hitung pemutakhiran bobot dari *input layer* ke *hidden layer* menggunakan Persamaan 2.11.

$$W = W'_{lama} - (learning\ rate \ \ dl_{dw}) \quad (2.11)$$

Keterangan:

W = nilai bobot baru dari *input layer* ke *hidden layer*
W'_{lama} = nilai bobot lama dari *input layer* ke *hidden layer*
Learning rate = parameter penurunan fungsi waktu
dl_{dw} = nilai bobot dari *input layer* ke *hidden layer*

    b. Hitung pemutakhiran bobot dari *hidden layer* ke *output layer* menggunakan Persamaan 2.12.

$$W' = W'_{lama} - (learning\ rate \ \ dl_{dw'}) \quad (2.12)$$

Keterangan:

W' = nilai bobot baru dari *hidden layer* ke *output layer*

W'_{lama} = nilai bobot lama dari *hidden layer* ke *output layer*
Learning rate = parameter penurunan fungsi waktu
dl_{dw} = nilai bobot dari *hidden layer* ke *output layer*

11. Hitung *query* dari masing-masing kata konteks untuk mendapatkan hasil berupa vektor kata menggunakan Persamaan 2.13.

$$\theta = \frac{v_{query}\ x\ (v_{konteks})^T}{\sqrt{\sum {v_{query}}^2}\ x\ \sqrt{\sum {v_{konteks}}^2}} \quad (2.13)$$

Keterangan:
θ = vektor kata
v_{query} = nilai bobot kata *query*
$v_{konteks}$ = nilai bobot kata konteks

12. Lakukan perangkingan nilai vektor kata konteks yang dihasilkan dari tiap kata mulai yang terbesar hingga terkecil. Selanjutnya untuk mengetahui kata konteks yang memiliki kedekatan dengan kata target, lakukan pengambilan kata dengan jumlah yang telah ditentukan.

2.8 *Cosine Similarity*

Cosine similarity merupakan salah satu metode yang sering diterapkan untuk proses temu kembali informasi dan klastering dimana akan diukur derajat kesamaan jarak antara dua dokumen (Lu, Shao, dan Yu, 2009). Untuk menghitung jarak *cosine similarity* pada dua dokumen dapat menggunakan Persamaan 2.14.

$$sim(d_j, q) = \frac{d_j \cdot q}{\|d_j\| \|q\|} = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2} \sqrt{\sum_{i=1}^{N} w_{i,q}^2}} \quad (2.14)$$

Keterangan :

$sim(d_j, q)$ = Similaritas dokumen ke-*j* dengan *query*
d_j = Dokumen
q = *Query*
$w_{i,j}$ = Bobot ke-*i* pada dokumen ke-*j*
$w_{i,q}$ = Bobot ke-*i* pada *query* ke-*j*

2.9 Evaluasi

Metode *Confusion Matrix* merupakan sebuah metode yang digunakan untuk mengevaluasi kinerja algoritme dan mengukur performa hasil rekomendasi dengan melakukan perhitungan nilai *Precision* (Manning, Schütze dan Raghavan, 2008). Dalam konteks rekomendasi, ketertarikan terhadap hasil yang diberikan memungkinkan untuk merekomendasi sebanyak item-item dengan hasil rekomendasi tertinggi sebanyak N (*Top-N*). Sehingga, pengujian dilakukan menggunakan *Precision@k* dan *Mean Average Precision* @k (MAP) dimana k merupakan hasil rekomendasi sebanyak *Top-N* yang akan dicocokkan dengan data penguji.

Nilai *Precision*@k adalah proporsi item yang direkomendasikan dalam set T*op-k* yang relevan, proses perhitungan nilai *Precision@k* akan mengambil bagian dokumen teratas sejumlah K dan mengabaikan dokumen yang berada dibawah K, dapat dilihat pada Persamaan 2.15.

1. $Precision = \frac{|\text{Relevan} \cap Retrieved|}{Retrieved}$ (2.15)

Keterangan:
Retrieved = Jumlah dokumen yang ditampilkan.
Relevan = Jumlah dokumen yang relevan/benar.

Nilai *Mean Average Precision* (MAP) adalah rata-rata dari nilai *precision* yang relevan dan memberikan nilai 0 pada item yang tidak relevan. Nilai MAP memperhatikan urutan yang dihasilkan oleh sistem. Persamaan 2.16 merupakan proses perhitungan nilai MAP.

2. $\text{MAP} = \frac{1}{\text{N}} \sum_{\text{j}=1}^{\text{N}} \frac{1}{\text{Q}_\text{j}} \sum_{\text{i}=1}^{\text{Q}_\text{j}} \text{Precision}(\text{Doc}_\text{i})$ (2.16)

Keterangan:
Q_j = Total dokumen relevan untuk *query j*
N = Total *query*
$Precision(\text{Doc}_\text{i})$ = Nilai *Precision* untuk dokumen relevan ke-*i*

BAB 3
METODOLOGI PENELITIAN

3.1 Tipe Penelitian

Penelitian rekomendasi film bertipe *non-implementatif* Analitik yaitu menitik beratkan terhadap hubungan-hubungan antar variabel pada sebuah permasalahan agar dapat menunjang pengambilan keputusan dalam rangka memenuhi objek penelitian yang sedang dilakukan.

3.2 Strategi Penelitian

Penelitian rekomendasi film memiliki strategi penelitian yang bersifat kuantitatif yaitu menggunakan perhitungan disetiap prosesnya. Penelitian rekomendasi film bersifat kuantitatif dikarenakan semakin banyak jumlah sinopsis yang digunakan untuk data *training* maka hasil rekomendasi akan semakin baik. Proses analisis dilakukan pada sinopsis film yang memiliki kedekatan jarak untuk menghasilkan sebuah rekomendasi film serupa. Sinopsis didapatkan dari hasil *crawling* data pada website *The Movie Database* (TMDb).

3.3 Partisipan Penelitian

Penelitian rekomendasi film melibatkan partisipan yaitu para penggemar film terutama penggemar film bahasa Inggris, dimana partisipan akan dilibatkan pada tahap *testing* untuk dapat mengukur keberhasilan proses perekomendasian film dengan memberikan tag relevan (R) atau tidak relevan (NR) terhadap rekomendasi yang dihasilkan sebanyak 10 film untuk tiap film yang dipilih. Jumlah partisipan yang dipakai untuk proses pengujian yaitu sebanyak 3 orang.

3.4 Lokasi Penelitian

Penelitian rekomendasi film memilih lokasi untuk melakukan penelitian yaitu pada lingkungan sekitar Universitas Brawijaya. Alasan pemilihan lokasi ini adalah agar mempermudah mendapatkan data penunjang seperti jurnal atau hasil penelitian yang sejenis.

3.5 Teknik Pengumpulan Data

Penelitian rekomendasi film menggunakan teknik pengumpulan data yaitu dengan melakukan *crawling* data yang diambil langsung dari halaman web *The Movie Database* (TMDb) mengenai film berbahasa Inggris.

3.6 Data Penelitian

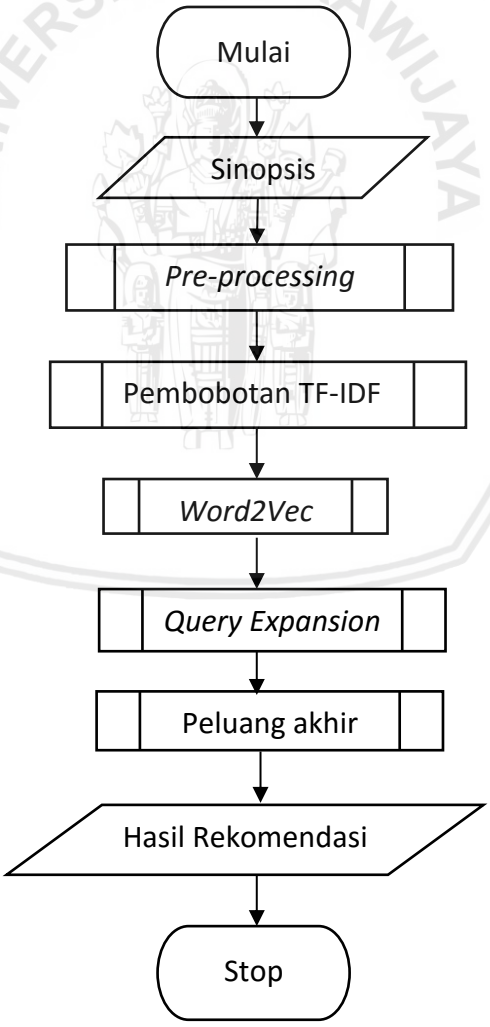
Penelitian rekomendasi film menggunakan data latih berupa judul dengan sinopsis film berjumlah 150 film berbahasa Inggris terbaru dari tahun 2000 sampai 2019. Proses evaluasi mengambil 30 data judul dan sinopsis dari data latih berdasarkan film yang dipilih oleh penguji.

3.7 Teknik Analisis Data

Penelitian rekomendasi film menggunakan teknik pengujian dari hasil perhitungan metode TF-IDF, *Word2Vec* maupun *Cosine Similarity* untuk menganalisa data. Pengujian dilakukan dengan menghitung nilai *Precision@k* dan *Mean Average Precision* (MAP).

3.8 Implementasi Algoritme

Penelitian rekomendasi film bedasarkan sinopsis menggunakan tahap *pre-processing* untuk memproses dokumen sinopsis film, metode TF-IDF untuk mendapatkan kata dengan bobot tertinggi sebanyak jumlah yang ditentukan bedasarkan hasil kecocokan *query* didalam dokumen, *Word2Vec* sebagai metode untuk mendapatkan *query expansion* dari hasil kata teratas yang diambil pada proses TF-IDF dan *cosine similarity* untuk mendapatkan nilai kemiripan dokumen dengan *query*. Hasil kemiripan dokumen dengan *query* dijadikan sebagai hasil rekomendasi yang diberikan kepada pengguna. Gambar 3.1 merupakan diagram alir proses rekomendasi film.



Gambar 3.1 Diagram alir proses rekomendasi

3.9 Peralatan Pendukung

Spesifikasi perangkat pendukung yang digunakan dalam penelitian rekomendasi film yaitu sebagai berikut:

a. Spesifikasi perangkat Lunak
   Perangkat lunak yang digunakan dalam penelitian ini, memiliki spesifikasi yang terdapat pada Tabel 3.1.

Tabel 3.1 Spesifikasi Perangkat Lunak

Nama komponen	**Spesifikasi**
Sistem operasi	Windows 10
Text editor	Spyder 3.6
Bahasa Pemrograman	Phyton
Library	Sklearn, NLTK, Numpy, Pickle, Pandas, BeautifulSoup

b. Spesifikasi perangkat keras
   Perangkat keras yang digunakan dalam penelitian ini, memiliki spesifikasi yang terdapat pada Tabel 3.2 Spesifikasi Perangkat Keras.

Tabel 3.2 Spesifikasi Perangkat Keras

Nama komponen	**Spesifikasi**
Processor	Intel(R) Core™ i3-M460 CPU @ 2.53GHz
Memori (RAM)	8.00 GB RAM
Chip Type	64-bit Operating System, x64-based processor

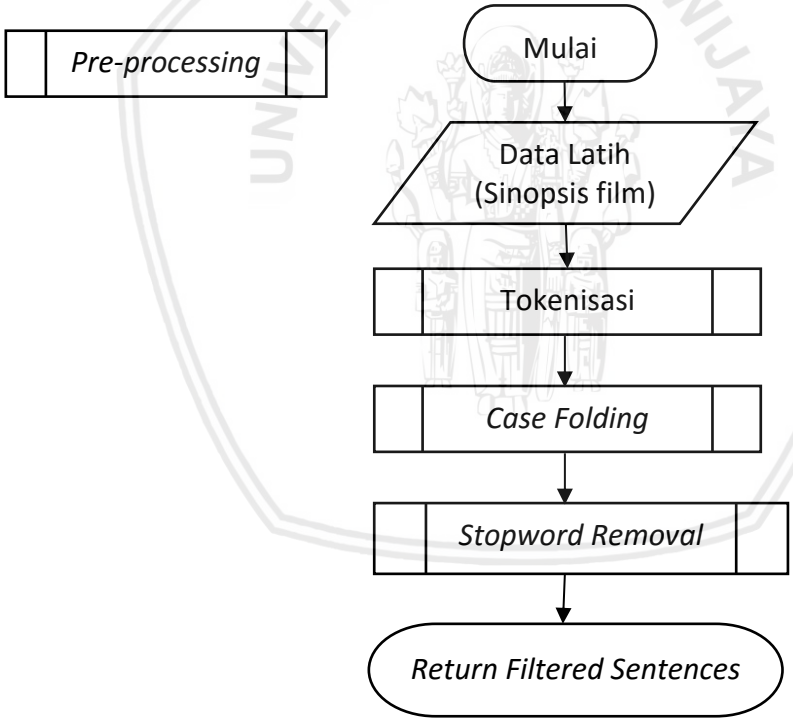
BAB 4
PERANCANGAN DAN IMPLEMENTASI

4.1 Diagram Alir Sistem

Diagram Alir atau *Flowchart* digunakan untuk mempermudah pemabahaman terhadap gambaran proses alir dari algoritme yang digunakan dalam proses rekomendasi. Proses yang akan dijelaskan yaitu *pre-processing*, metode *Word2Vec*, pembuatan *Query Expansion* dan hasil akhir rekomendasi dengan *Cosine Similarity*. Penjelasan untuk masing-masing proses terdapat dalam subbab 4.1.1 sampai subbab 4.1.5.

4.1.1 *Pre-processing*

Pre-processing merupakan tahapan awal untuk membersihkan data dari kata yang tidak memiliki arti . Tahapan dalam *pre-processing* yaitu terdiri dari tokenisasi dan *case folding* dan *stopword removal.* Tahap ini bertujuan membuat term atau kata dari sinopsis. Tahap dari *pre-processing* terdapat pada Gambar 4.1.

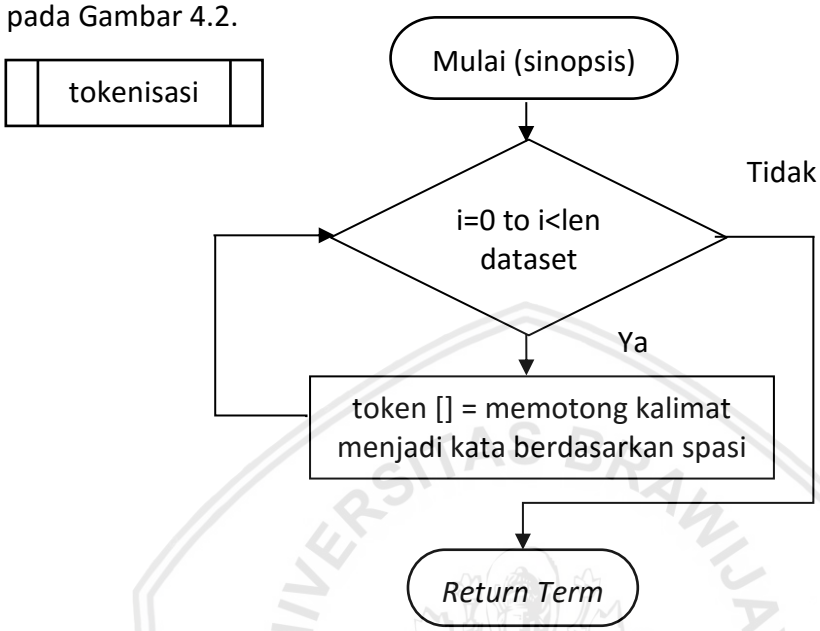


Gambar 4.1 Diagram Alir *Pre-processing*

Diagram alir pada Gambar 4.1 merupakan proses *pre-processing*. Proses *pre-processing* yang digunakan yaitu tokenisasi, *case folding*, dan *stopword removal*. Data latih yang digunakan berupa sinopsis dari 150 film berbahasa Inggris, kumpulan sinopsis tersebut dilakukan proses tokenisasi yaitu pemotongan kalimat menjadi kumpulan kata, selanjutnya mengkonversi seluruh kata menjadi bentuk *lowercase* atau huruf kecil, serta penghilangan kata-kata yang termasuk ke dalam *stopword* (kata yang tidak penting). Hasil dari proses *pre-processing* berupa kumpulan kata/term yang lebih terstruktur agar lebih mudah untuk diolah.

4.1.1.1 Tokenisasi

Tokenisasi merupakan tahap untuk memecah kalimat sinopsis menjadi beberapa kata serta menghapus karakter selain huruf untuk proses selanjutnya. Hasil dari proses tokenisasi adalah kumpulan term. Tahap dari tokenisasi terdapat pada Gambar 4.2.

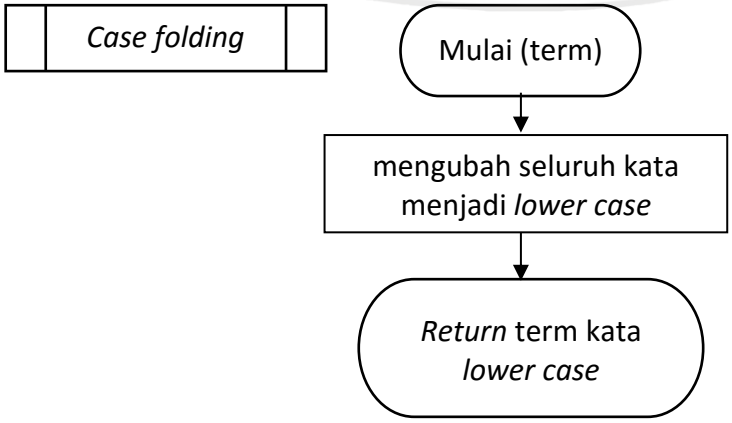


Gambar 4.2 Diagram Alir Tokenisasi

Diagram alir pada Gambar 4.2 merupakan proses tokenisasi. Tokenisasi memproses seluruh dokumen yang ada pada data latih, dimulai pada data latih pertama sampai sejumlah 150. Proses tokenisasi akan memotong kalimat menjadi kumpulan kata/term berdasarkan spasi. Akhir dari proses tersebut berupa kumpulan kata/term.

4.1.1.2 *Case folding*

Tahap *case folding* bertujuan untuk mengubah seluruh kata dalam dokumen menjadi *lowercase*. Proses dari *case folding* terdapat pada Gambar 4.3.

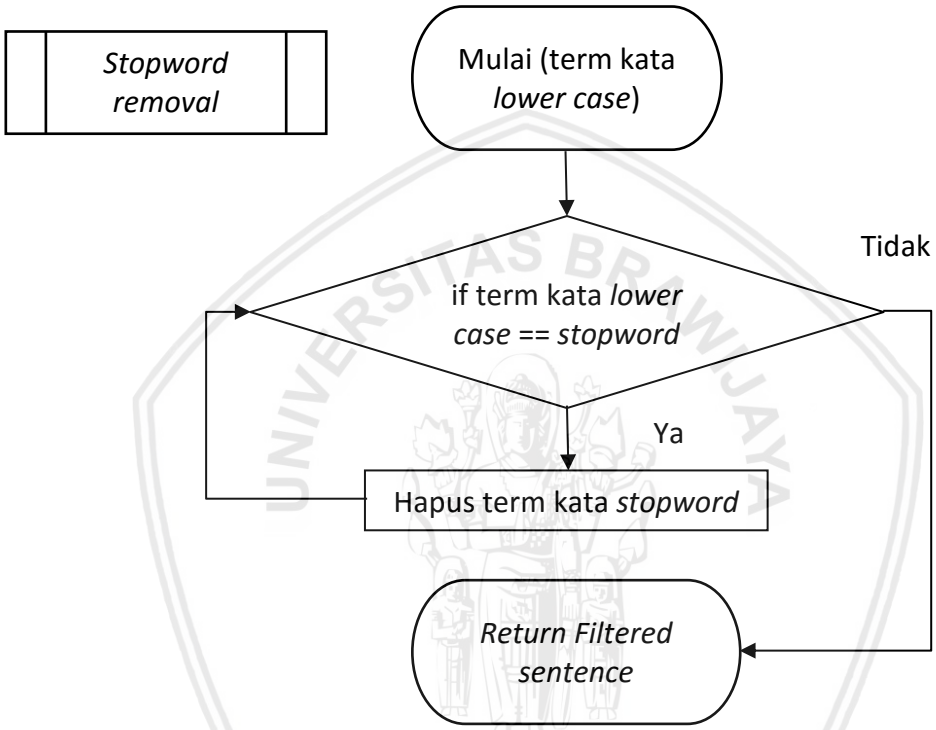


Gambar 4.3 Diagram Alir *Case folding*

Diagram alir pada Gambar 4.3 merupakan proses *case folding*. Kumpulan term hasil dari proses tokenisasi selanjutnya diubah menjadi bentuk *lowercase* atau diubah menjadi huruf kecil. Akhir dari proses tersebut berupa kumpulan kata/term *lowercase*.

4.1.1.3 *Stopword Removal*

Stopword removal merupakan suatu proses pembuangan kata yang tidak penting. *Stopword Removal* dibuat menggunakan bantuan *library* NLTK Porter Stemmer. Proses dari *stopword removal* terdapat pada Gambar 4.4.

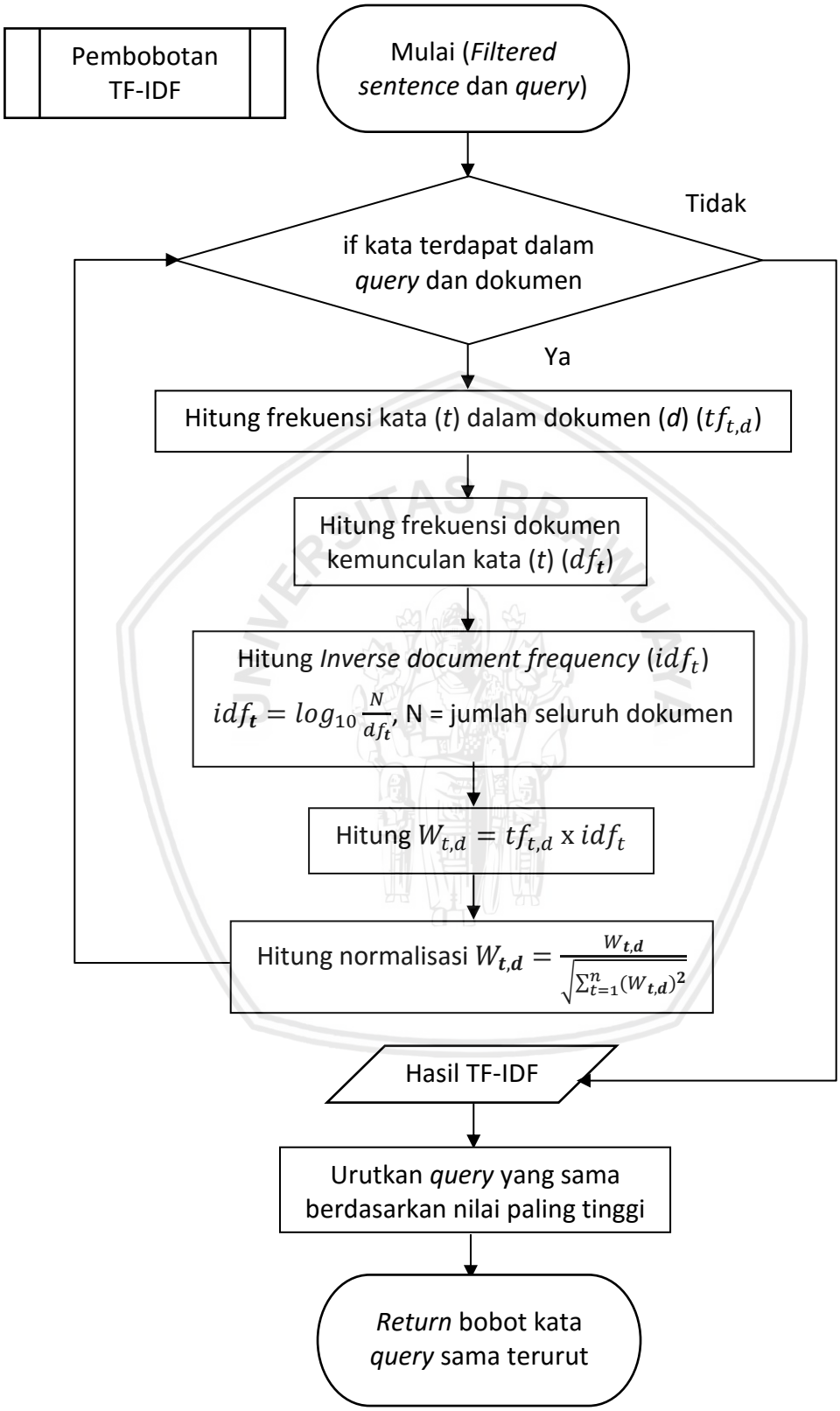


Gambar 4.4 Diagram Alir *Stopword Removal*

Diagram alir pada Gambar 4.4 merupakan proses *stopword removal*. Setelah dilakukan proses tokenisasi dan *case folding*, kumpulan kata/ term yang sudah berbentuk *lowercase* selanjutnya dilakukan proses *stopword removal*. *Stopword removal* merupakan suatu proses pembuangan kata yang sering muncul namun memiliki arti yang tidak relevan sehingga tidak mampu mewakili sebuah dokumen. Hasil dari proses ini yaitu kata/ term yang sudah bersih dari kata *stopword*.

4.1.2 Pembobotan TF-IDF

Tujuan dari adanya pembobotan kata agar setiap kata diberi bobot sesuai frekuensi kemunculkan kata dalam suatu dokumen. Proses pembuatan bobot dengan menggunakan TF-IDF terdapat pada Gambar 4.5.

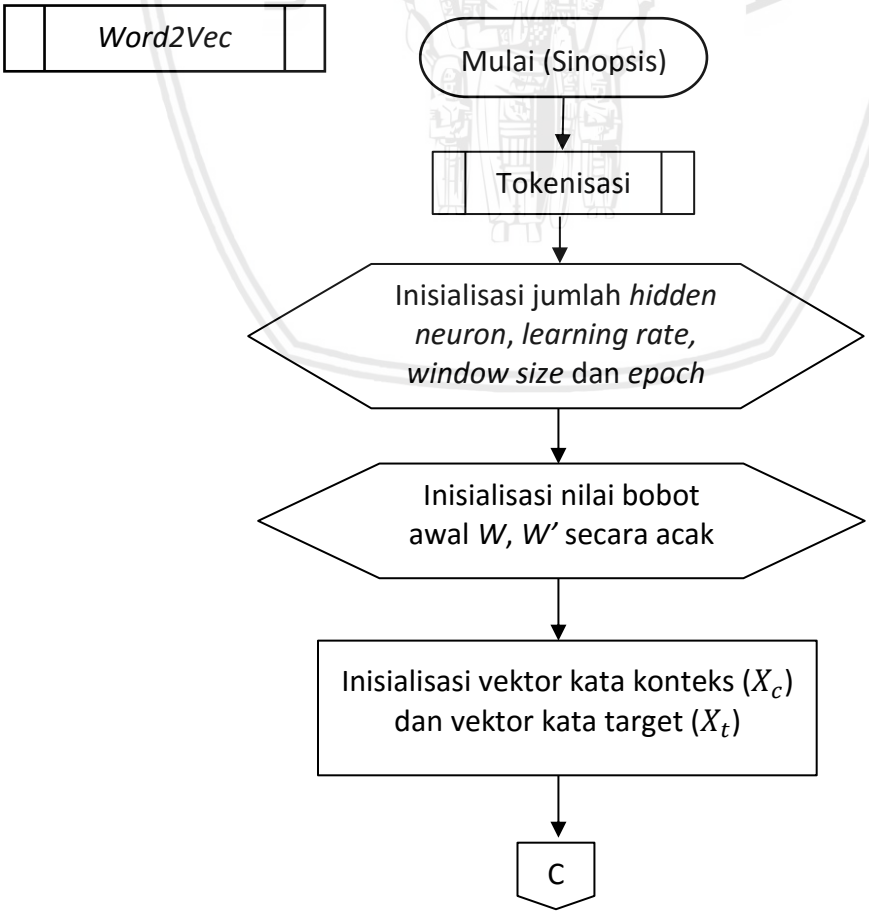


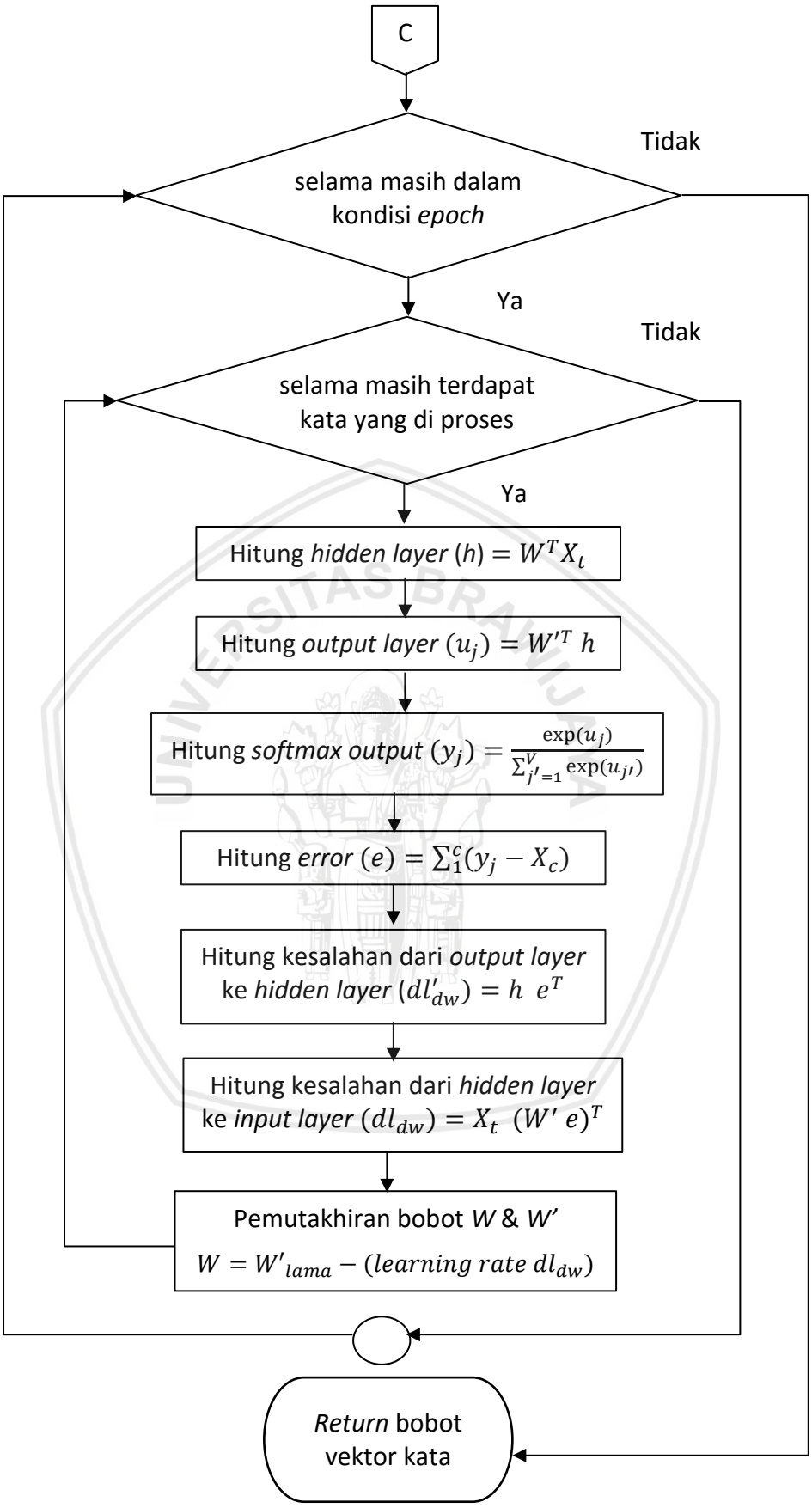
Gambar 4.5 Diagram Alir TF-IDF

Diagram alir pada Gambar 4.5 merupakan proses pembuatan bobot dengan menghitung *Term Frequency-Inverse Document Frequency* (TF-IDF). Proses perhitungan dilakukan pada seluruh kata yang telah melalui proses *pre-processing* untuk semua dokumen maupun *query input*-an pengguna. Proses pertama yaitu menghitung frekuensi kemunculan kata/term (*tf*) pada sebuah dokumen, proses dilakukan menggunakan teknik *one hot encoding* dengan kondisi seperti yang tertera pada Persaamaan 2.1. Setelah mendapatkan frekuensi kemunculan kata, hitung frekuensi dokumen dimana suatu kata/term muncul (*df*) dan hitung inverse dari nilai df tersebut (*idf*) menggunakan rumus pada persamaan 2.2. Nilai bobot yang dihasilkan dari proses TF-IDF merupakan perkalian nilai *tf* dan *idf.* Selanjutnya, untuk menormalkan nilai bobot yang dihasilkan, hitung normalisasi menggunakan rumus pada persamaan 2.4. Hasil nilai bobot kata yang ada pada *query* diurutkan berdasarkan nilai yang paling tinggi. Hasil dari proses ini yaitu mendapatkan nilai bobot kata-kata *query* terurut berdasarkan nilai tertinggi yang akan diambil sebanyak *Top-N* kata untuk mendapat penambahan *query*.

4.1.3 Metode *Word2Vec*

Metode *Word2Vec* akan memproses kata perkata dalam sebuah kalimat secara terurut pada seluruh dokumen. Proses perhitungan *Word2Vec* dilakukan untuk menghasilkan bobot seluruh kata vocab yang terdapat dalam dokumen. Nilai bobot yang dihasilkan digunakan untuk proses *query expansion*. Gambar 4.6 menunjukkan tahapan proses *Word2Vec.*



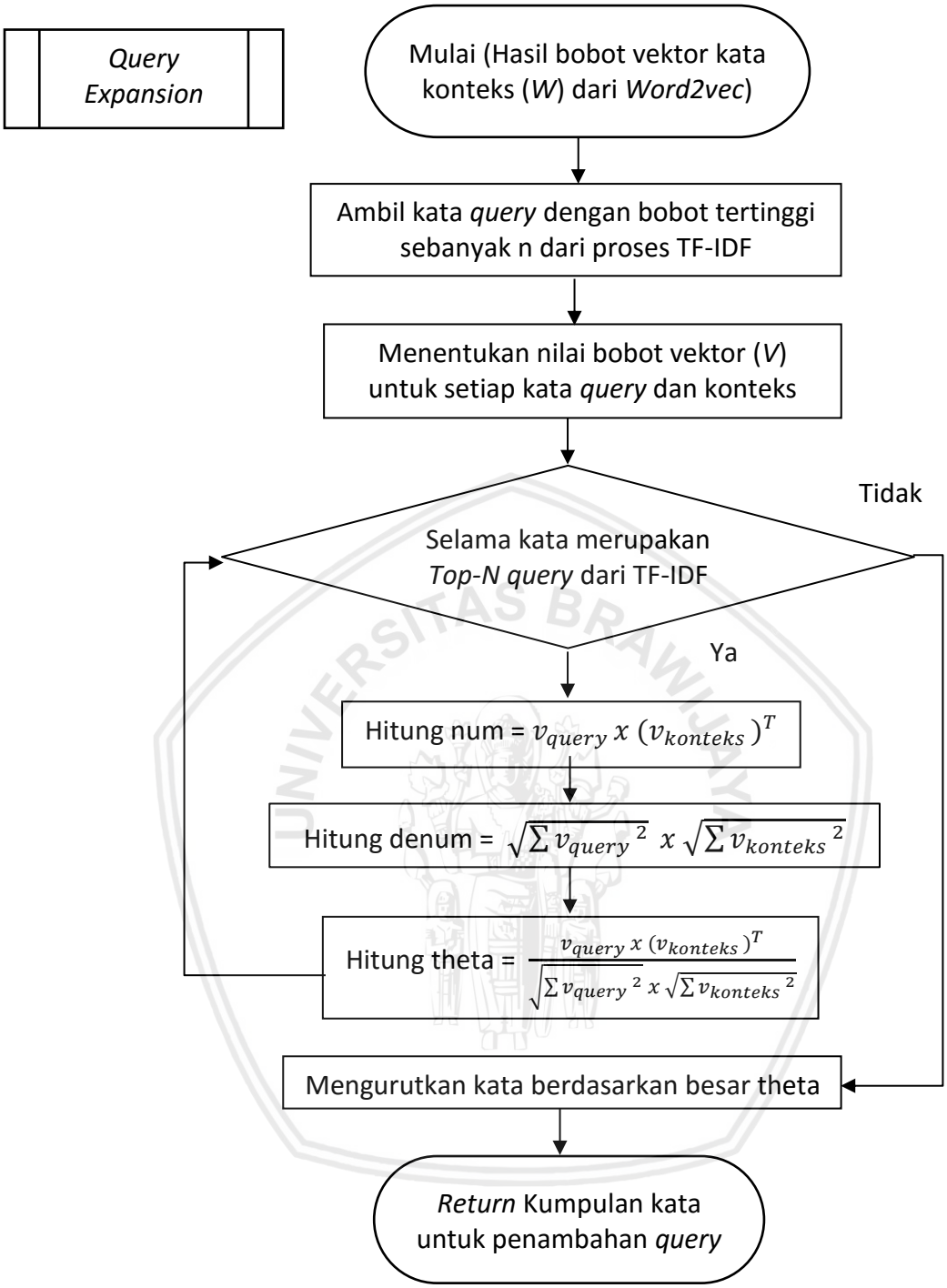


Gambar 4.6 Diagram Alir *Word2Vec*

Diagram alir pada Gambar 4.6 merupakan proses pembuatan matriks bobot dengan metode *Word2Vec*. Proses perhitungan dilakukan untuk kata perkata dalam sebuah kalimat secara terurut pada seluruh dokumen, data yang diproses berupa data latih sinopsis dari 150 film berbahasa inggris. Proses pertama yaitu melakukan inisialisasi jumlah *hidden neuron*, *learning rate*, *window size* dan jumlah *epoch*. Karena proses dilakukan kata perkata maka, lakukan proses tokenisasi yaitu pemotongan kalimat menjadi kata/term. Setelah seluruh kalimat yang ada pada data latih telah menjadi potongan kata, selanjutnya membuat vektor kata konteks dan kata target menggunakan teknik *one hot encoding*, dimana kata akan bernilai 1 apabila kata tersebut merupakan representasi kata yang dipilih dan kata lainnya akan bernilai 0. Selanjutnya melakukan proses inisialisasi nilai bobot konteks dan embedding secara acak, ukurannya melibatkan jumlah *hidden neuron* dan jumlah *vocab*. Terdapat kondisi perulangan yaitu selama masih dalam kondisi *epoch* maka akan mengecek kondisi lagi apabila kata tersebut merupakan kata target maka akan menghitung nilai bobot dari *input layer* ke *hidden layer* (*h*) menggunakan rumus pada Persamaan 2.5. Selanjutnya, hitung juga nilai bobot dari *hidden layer* ke *output layer* (u_j) menggunakan persamaan 2.6. Dalam mendapatkan hasil prediksi, nilai bobot dari dari *hidden layer* ke *output layer* (u_j) dihitung fungsi aktivasinya menggunakan rumus *softmax output* (y_j) pada persamaan 2.7. Setelah itu, untuk menghitung nilai error hasil prediksi, hitung nilai *error* (*e*) dari nilai *softmax* baris ke *j* dikurangi nilai vektor kata konteks baris ke *i* menggunakan Persamaan 2.8. Selanjutnya, untuk meminimalisir kesalahan, hitung *backpropagation* dari nilai bobot *output layer* ke *hidden layer* (dl'_{dw}) menggunakan Persamaan 2.9 dan nilai bobot dari *hidden layer* ke *input layer* (dl_{dw})menggunakan Persamaan 2.10. Proses selanjutnya, hitung pemutakhiran bobot (*update* bobot) untuk mengubah nilai bobot dari bobot awal matriks konteks (W) dan matriks *embedding* (W') menggunakan rumus pada Persamaan 2.10. Nilai yang dihasilkan dari pemutakhiran bobot akan digunakan untuk nilai bobot pada proses kata berikutnya, selanjutnya pengecekan looping apabila seluruh kata target sudah di proses dan kondisi epoch sudah terpenuhi maka menghasilkan nilai bobot akhir matriks konteks (W) dan matriks *embedding* (W'). Nilai bobot akhir dari matriks konteks (W) yang akan dipakai untuk melakukan proses *query expansion.*

4.1.4 Pembuatan *Query Expansion*

Proses *Query Expansion* dihasilkan dengan menghitung nilai num, denum dan tetha berdasarkan nilai bobot vektor konteks terakhir dari proses *Word2Vec*. Penambahan *query* baru akan diambil nilai tertinggi bedasarkan jumlah yang ditentukan untuk ditambahkan ke *query* awal hasil TF-IDF. Tahap dari *Query Expansion* terdapat pada Gambar 4.7.



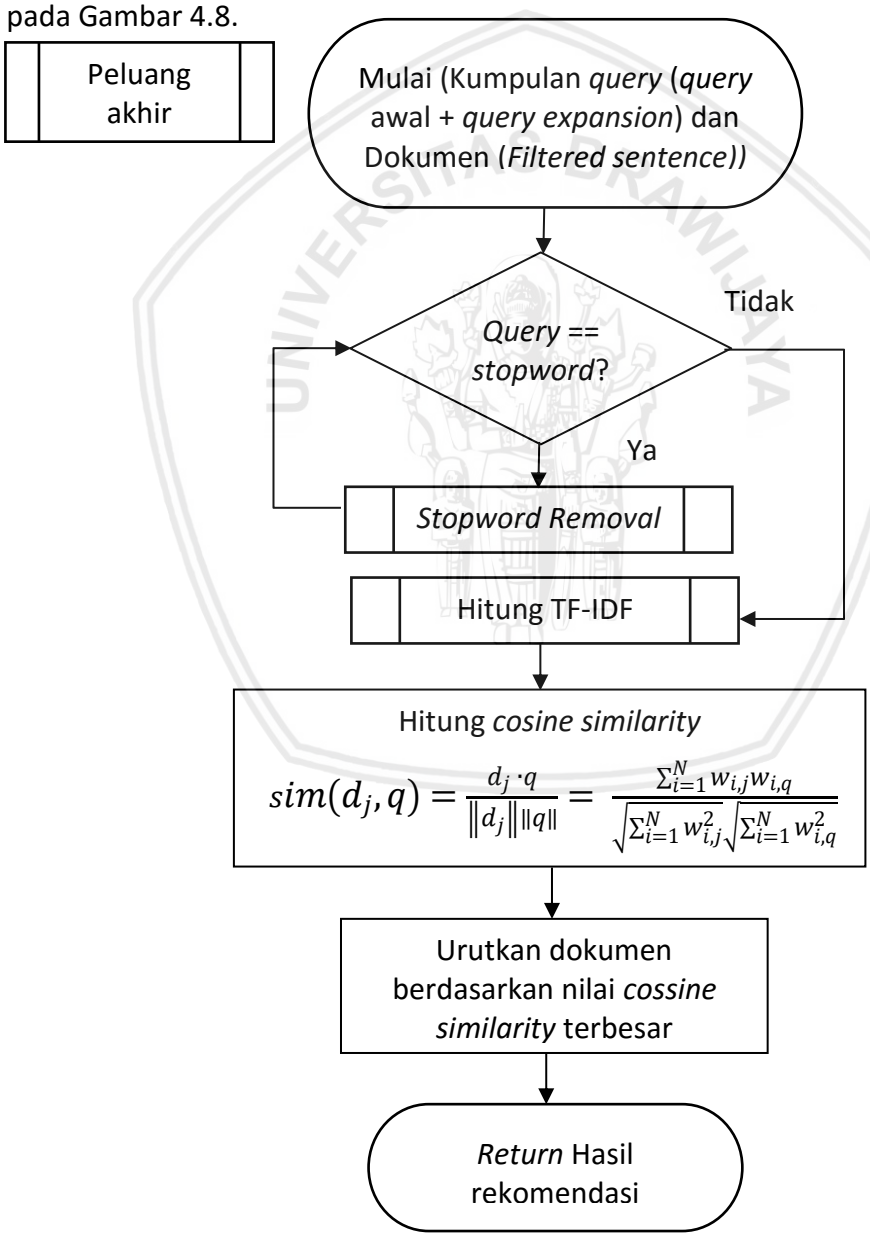
Gambar 4.7 Diagram Alir *Query Expansion*

Diagram alir pada Gambar 4.7 merupakan proses pembuatan *Query Expansion* menggunakan nilai bobot matriks konteks hasil dari metode *Word2Vec*. Proses perhitungan dilakukan untuk mencari jarak terdekat antara kata *query* dengan seluruh kata konteks, kata *query* merupakan kata-kata dengan nilai TF-IDF tertinggi yang diambil sebanyak n, sedangkan kata konteks merupakan seluruh kata yang ada pada data latih. Kata-kata *query* tersebut yang akan mendapat penambahan *query*. Selajutnya, mencari nilai bobot vektor (*V*) untuk masing-masing kata *query* maupun kata konteks, nilai bobot yang digunakan yaitu hasil dari proses *Word2Vec.* Setelah didapat nilai bobot vektor kata *query* (v_{query}) dan konteks ($v_{konteks}$), proses

dilanjutkan dengan perhitungan nilai num, denum dan theta menggunakan Persamaan 2.13 untuk setiap kata pada matriks nilai *V.* Nilai theta dari setiap kata *query* yang didapat, diurutkan berdasarkan nilai tertinggi. Pada akhir proses *Query Expansion* diambil kata-kata yang akan ditambahkan ke *query* awal berdasarkan dari nilai theta yang tertinggi, semakin tinggi nilai tetha maka semakin mirip dengan kata *query*. Hasil dari proses ini yaitu berupa kumpulan kata hasil ekspansi *query* yang akan ditambahkan ke dalam *query* awal.

4.1.5 Peluang Akhir untuk menghasilkan Rekomendasi

Dalam menghasilkan proses rekomendasi yang sesuai, lakukan perhitungan nilai kemiripan antara *query* dengan dokumen menggunakan algoritme *cosine similarity*. Tahap menghitung peluang hasil akhir proses rekomendasi dapat dilihat pada Gambar 4.8.



Gambar 4.8 Diagram Alir Peluang hasil akhir rekomendasi

Diagram alir pada Gambar 4.8 merupakan proses perhitungan peluang akhir hasil rekomendasi menggunakan metode *cosine similarity*. Proses perhitungan peluang akhir diawali dengan mendefinisikan kumpulan *query* dan juga dokumen. Dilanjutkan dengan perulangan untuk membandingkan apakah *query* tersebut merupakan *stopword*, apabila query tersebut sama dengan *stopword* maka akan dilanjutkan dengan proses *stopword removal* atau penghapusa kata, namun apabila kata tersebut bukan termasuk kata *stopword* maka dilanjutkan pada proses pembuatan bobot dengan perhitungan TF-IDF untuk mendapatkan nilai bobot *query* baru setelah mendapat *query* tambahan. Setelah mendapatkan nilai bobot baru hasil dari perhitungan TF-IDF dilanjutkan dengan mencari nilai kedekatan antara *query* dengan seluruh dokumen yang ada dalam data latih menggunakan perhitungan *cosine similarity* dengan rumus pada Persamaan 2.14. Nilai dari hasil perhitungan jarak antara *query* dengan seluruh dokumen menggunakan *cosine similarity,* dilakukan proses pengurutan nilai dari yang tertinggi hingga nilai terendah untuk menentukan dokumen yang memiliki kemiripan terdekat dan akan ditampilkan sebagai hasil rekomendasi yang diberikan sistem. Hasil dengan nilai *cosine* tertinggi merupakan hasil dari film yang paling dekat atau mirip dengan film dari *query* pilihan pengguna.

4.2 Manualisasi

Permasalahan yang akan diselesaikan pada penelitian ini yaitu pembuatan rekomendasi film berdasarkan sinopsis. Data yang digunakan untuk proses manualisasi terdiri dari 3 kalimat untuk data latih dan 1 kalimat untuk data uji, dalam 1 dokumen melambangkan sebuah kalimat yang terdapat pada sinopsis dari sebuah film. Terdapat beberapa tahapan dalam proses manualisasi, antara lain *Pre-processing*, pembuatan bobot dengan TF-IDF, pembuatan matriks bobot dengan *Word2ec*, pembuatan *Query Expansion* dan pencarian nilai similaritas dengan *Cosine Similarity* untuk mendapatkan hasil akhir. Sampel data latih dapat dilihat pada Tabel 4.1 sedangkan data uji dapat dilihat pada Tabel 4.2.

Tabel 4.1 Data Latih

Dokumen	Judul film	Sinopsis (kalimat)
1.	The Conjuring	Paranormal investigators
2.	The Conjuring 2	malicious spirits
3.	IT	bullies and a monster that takes the shape of a clown

Tabel 4.2 Data Uji

Dokumen	Judul film	Sinopsis (kalimat)
1.	Clown	father finds a clown suit for his son's birthday

4.2.1 *Pre-processing*

Langkah pertama yaitu melakukan *pre-processing* data, terdiri dari tokenisasi, *case folding* dan *stopword removal*. Tabel 4.3 menunjukkan contoh hasil dari *pre-procesing* data latih.

Tabel 4.3 *Pre-processing*

Doc	Judul film	Sinopsis	Tokenisasi	*Case folding*	*Stopword removal*
1	The Conjuring	Paranormal investigators	- Paranormal - investigator	- paranormal - investigator	- paranormal - investigator
2	The Conjuring 2	malicious spirits	- malicious - spirits	- malicious - spirits	- malicious - spirits
3	IT	bullies and a monster that takes the shape of a clown	- bullies - and - a - monster - that - takes - the - shape - of - a - clown	- bullies - and - a - monster - that - takes - the - shape - of - a - clown	- bullies - monster - takes - shape - clown

4.2.2 Pembuatan Nilai Bobot dengan TF-IDF

Pada tahap ini, mencari nilai bobot awal setiap kata yang terdapat dalam dokumen. Perhitungan dilakukan dengan menghitung nilai *Term frequency-invers document frequency* (TF-IDF). Kata yang dipakai untuk perhitungan nilai bobot merupakan kata yang sudah dilakukan proses tokenisasi, *case folding* serta pembuangan kata *stopword*. Selanjutnya masuk ke tahap perhitungan nilai *Term frequency-invers document frequency* (TF-IDF), langkah pertama yaitu menghitung frekuensi kemunculan kata pada setiap dokumen dan *query*. Kalimat dalam data uji dianggap sebagai *query* masukkan, maka kata dari *query* yang terdapat didalam dokumen hanya kata "clown". Tabel 4.4 menunjukkan jumlah frekuensi kemunculan kata yang muncul pada setiap dokumen dan *query*.

Tabel 4.4 Frekuensi kemunculan kata

Term	D1	D2	D3	*Query*
paranormal	1	0	0	0
investigators	1	0	0	0
malicious	0	1	0	0
spirits	0	1	0	0
bullies	0	0	1	0
monster	0	0	1	0
takes	0	0	1	0
shape	0	0	1	0
clown	0	0	1	1

Langkah kedua yaitu menghitung *term frecuency* atau frekuensi kata menggunakan rumus pada persamaan 2.1. Berikut contoh perhitungan manual untuk menghitung nilai *term frequency* dari kata paranormal pada dokumen 1.

$$\mathrm{tf}_{t,d} = 1 + \log_{10} 1 = 1$$

Tabel 4.5 menunjukkan nilai dari *term frecuency.*

Tabel 4.5 *Term frequency* ($\mathrm{tf}_{t,d}$)

Term	D1	D2	D3	*Query*
paranormal	1	0	0	0
investigators	1	0	0	0
malicious	0	1	0	0
spirits	0	1	0	0
bullies	0	0	1	0
monster	0	0	1	0
takes	0	0	1	0
shape	0	0	1	0
clown	0	0	1	1

Langkah ketiga yaitu menghitung *document frecuency* (df_t) atau frekuensi dokumen yang menyatakan banyaknya dokumen dimana suatu kata/term/token muncul. Tabel 4.6 menunjukkan contoh nilai dari *document frecuency.*

Tabel 4.6 *Document frequency* ($\mathbf{tf_{t,d}}$)

TERM	dft
paranormal	1
investigators	1
malicious	1
spirits	1
bullies	1
monster	1
takes	1
shape	1
clown	1

Langkah keempat yaitu menghitung *inverse document frecuency* ($\mathrm{idf_t}$) dengan rumus pada persamaan 2.2. Berikut contoh perhitungan manual untuk *inverse document frecuency* ($\mathrm{idf_t}$) pada kata paranormal dengan rumus pada persamaan 2.2.

$$idf_{\boldsymbol{t}} = log_{10}\frac{3}{1} = log_{10}\,3 = 0{,}47712125$$

Tabel 4.7 menunjukkan contoh nilai dari *inverse document frecuency* seluruh kata.

Tabel 4.7 *Inverse document frequency* ($\mathbf{idf_t}$)

TERM	N/dft	idft
paranormal	3	0,477121
investigators	3	0,477121
malicious	3	0,477121
spirits	3	0,477121
bullies	3	0,477121
monster	3	0,477121
takes	3	0,477121
shape	3	0,477121
Clown	3	0,477121

Langkah kelima yaitu menghitung *frequency-invers document frequency* (TF-IDF) dengan rumus pada persamaan 2.3. Berikut contoh perhitungan manual untuk *frequency-invers document frequency* ($\mathrm{W_{t,d}}$) pada kata "clown" yang ada di dalam *query* menggunakan rumus pada persamaan 2.3.

$$W_{t,d}(clown) = 1 \text{ x } 0{,}47712 = 0{,}47712$$

Tabel 4.8 menunjukkan nilai *frequency-invers document frequency* (TF-IDF) yang dihasilkan dari seluruh kata.

Tabel 4.8 *Frequency-invers document frequency* ($W_{t,d}$)

TERM	D1	D2	D3	*Query*
paranormal	0,477121	0	0	0
investigators	0,477121	0	0	0
malicious	0	0,477121	0	0
spirits	0	0,477121	0	0
bullies	0	0	0,477121	0
monster	0	0	0,477121	0
takes	0	0	0,477121	0
shape	0	0	0,477121	0
clown	0	0	0,477121	0,477121

Langkah keenam yaitu melakukan perhitungan normalisasi data dengan rumus pada persamaan 2.4. Berikut contoh perhitungan manual normalisasi data pada kata "clown" yang ada di dalam dokumen menggunakan rumus pada persamaan 2.4.

$$W_{t,d} = \frac{0{,}477121}{\sqrt{0{,}477121^2}} = \frac{0{,}477121}{0{,}227645} = 1$$

Tabel 4.9 menunjukkan contoh nilai dari hasil normalisasi seluruh kata yang ada dalam dokumen.

Tabel 4.9 Hasil Normalisasi

TERM	D1	D2	D3	*Query*
paranormal	0,70711	0	0	0
investigators	0,70711	0	0	0
malicious	0	0,70711	0	0
spirits	0	0,70711	0	0
bullies	0	0	0,477121	0
monster	0	0	0,477121	0
takes	0	0	0,477121	0
shape	0	0	0,477121	0
clown	0	0	0,477121	1

Dari hasil perhitungan *frequency-invers document frequency* (TF-IDF) Normalisasi yang dihasilkan dari seluruh kata, dilakukan proses pengurutan dari nilai tertinggi sampai terendah. Selanjutnya, ditetapkan jumlah kata yang ingin

diambil untuk dilakukan proses *Query Expansion* dengan metode *Word2Vec*. Apabila diambil 1 kata dengan nilai tertinggi, maka kata yang diambil berupa kata "clown".

4.2.3 Pembuatan Matriks Bobot *Word2Vec*

Perhitungan metode *Word2Vec* dilakukan untuk semua kata pada dokumen yang terdapat dalam data latih. Sebelum melakukan perhitungan, lakukan proses *pre-processing* data yaitu tokenisasi agar mendapatkan kumpulan kata dari suatu dokumen tersebut. Proses tokenisasi dapat dilihat pada Tabel 4.3. Proses perhitungan pada metode *Word2Vec* dilakukan pada setiap kata. *Word2Vec* memiliki konsep yaitu kata target yang dikelilingi oleh kata-kata konteks. Pada manualisasi ini menggunakan konsep *window* 1, memiliki arti yaitu kata yang diambil hanya satu kata dari sebelah kanan dan atau satu kata dari sebelah kiri kata target. Proses akan bergeser dari sebelah kiri ke kanan. Tabel 4.10, Tabel 4.11 dan Tabel 4.12 menunjukkan gambaran penentuan kata target dan kata konteks dari kalimat pada dokumen 1, dokumen 2 dan dokumen 3.

Tabel 4.10 Penentuan kata target dan kata konteks dokumen 1

Paranormal	investigators
Paranormal	
Paranormal	investigators
	investigators

Tabel 4.11 Penentuan kata target dan kata konteks dokumen 2

malicious	spirits
malicious	
malicious	spirits
	spirits

Tabel 4.12 Penentuan kata target dan kata konteks dokumen 3

Bullies	and	a	monster	that	takes	the	shape	of	a	Clown
Bullies	and									
Bullies	and	a								
	and	a	monster							
		a	monster	that						
			monster	that	takes					
				that	takes	the				
					takes	the	shape			
						the	shape	of		

Tabel 4.12 Penentuan kata target dan kata konteks dokumen 3 (lanjutan)

Bullies	and	a	monster	that	takes	the	shape	of	a	Clown
							shape	of	a	
								of	a	Clown
									a	Clown

4.2.3.1 Inisialisasi Awal

Tahap pertama sebelum melakukan perhitungan dengan menggunakan metode *Word2Vec* yaitu menentukan jumlah *hidden neuron*, *window size* serta nilai *learning rate*. Dalam perhitungan manual ini, jumlah *hidden neuron* sebanyak 3, *window size* sebanyak 1 dan nilai *learning rate* sebesar 0,01.

4.2.3.2 Inisialisasi Vektor Kata

Melakukan inisialisasi nilai vektor dari kata target dan vektor kata konteks dengan menggunakan teknik *one hot encoding* dimana kata akan bernilai 1 apabila kata tersebut merupakan representasi kata yang dipilih dan kata lainnya akan bernilai 0. Untuk melakukan inisialisasi vektor kata, kumpulan kata diurutkan terlebih dahulu berdasarkan abjad, dan setiap kata target ataupun kata konteks memiliki vektor dengan panjang sebanyak jumlah kata. Tabel 4.13 menunjukkan nilai dari vektor kata target (w(t)) dan vektor kata konteks (w(c)) dari kata "Paranormal".

Tabel 4.13 Vektor kata target dan vektor kata konteks

VOCAB	w(t)	w(c)
a	0	1
and	0	0
bullies	0	0
clown	1	0
investigator	0	1
malicious	0	0
monster	0	0
of	0	0
Paranormal	1	0
shape	0	0
spirits	0	0
takes	0	0
that	0	0
the	0	0

4.2.3.3 Inisialisasi W dan W'

Melakukan inisialisasi nilai bobot awal matriks konteks (*W*) dan matriks *embedding* (*W'*) dengan membangkitkan nilai secara acak. Nilai bobot dari matriks konteks (W) berupa matriks ordo 14X3 dimana 14 merupakan jumlah kata dan 3 merupakan jumlah *hidden neuron*. Sedangkan untuk nilai bobot dari matriks *embedding* (*W'*) berupa matriks ordo 3X14 dimana 3 merupakan jumlah *hidden neuron* dan 14 merupakan jumlah kata. Tabel 4.14 menunjukkan nilai bobot matriks konteks (*W*) dan Tabel 4.15 menunjukkan nilai bobot matriks *embedding* (*W'*) dari kata "Paranormal".

Tabel 4.14 Matriks *W*

0,5	0,3	0,5
0,1	-0,6	0,7
0,4	0,8	0,2
-0,6	0,3	-0,1
0,5	0,2	-0,4
0,7	0,6	-0,3
-0,2	0,4	0,1
0,1	0,8	0,2
0,4	0,1	0,3
-0,2	0,3	0,6
0,2	0,8	0,1
-0,1	-0,6	0,4
-0,6	0,5	0,2
-0,1	0,3	0,4

Tabel 4.15 Matriks *W'*

0,8	-0,2	0,3	-0,1	0,7	0,9	-0,6	0,1	0,4	0,3	-0,8	0,1	0,7	0,3
0,4	0,2	0,8	0,1	0,3	0,1	0,6	0,6	0,8	-0,2	0,3	0,2	0,1	0,1
0,1	0,6	0,1	-0,7	0,7	0,2	0,5	0,2	-0,4	0,7	-0,4	0,5	-0,4	-0,7

4.2.3.4 *Forward Pass*

Setelah data teks diubah menjadi representasi biner dengan teknik *one hot encoding*, selanjutnya melakukan perhitungan dari *input layer* ke *hidden layer* dan perhitungan dari *hidden layer* ke *output layer*. Untuk mendapatkan nilai matriks dari perhitungan *input layer* ke *hidden layer*, dilakukan dengan cara mengalikan matriks target (w(t)), dengan matriks bobot (*W*) yang telah di transposisi. Tabel 4.16 menunjukkan nilai bobot matriks konteks (*W*) yang telah di transposisi.

Tabel 4.16 Matriks W^T

0,5	0,1	0,4	-0,6	0,5	0,7	-0,2	0,1	0,4	-0,2	0,2	-0,1	-0,6	-0,1
0,3	-0,6	0,8	0,3	0,2	0,6	0,4	0,8	0,1	0,3	0,8	-0,6	0,5	0,3
0,5	0,7	0,2	-0,1	-0,4	-0,3	0,1	0,2	0,3	0,6	0,1	0,4	0,2	0,4

Berikut contoh perhitungan manual untuk mendapatkan nilai dari *input layer* ke *hidden layer* pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.5.

$$h = \begin{bmatrix} 0{,}5 & 0{,}1 & 0{,}4 & -0{,}6 & 0{,}5 & 0{,}7 & -0{,}2 & 0{,}1 & 0{,}4 & -0{,}2 & 0{,}2 & -0{,}1 & -0{,}6 & -0{,}1 \\ 0{,}3 & -0{,}6 & 0{,}8 & 0{,}3 & 0{,}2 & 0{,}6 & 0{,}4 & 0{,}8 & 0{,}1 & 0{,}3 & 0{,}8 & -0{,}6 & 0{,}5 & 0{,}3 \\ 0{,}5 & 0{,}7 & 0{,}2 & -0{,}1 & -0{,}4 & -0{,}3 & 0{,}1 & 0{,}2 & 0{,}3 & 0{,}6 & 0{,}1 & 0{,}4 & 0{,}2 & 0{,}4 \end{bmatrix} x \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0{,}4 \\ 0{,}1 \\ 0{,}3 \end{bmatrix}$$

Sedangkan, untuk mendapatkan nilai matriks dari perhitungan *hidden layer* ke *output layer*, dilakukan dengan cara mengalikan hasil perhitungan *input layer* ke *hidden layer* (*h*) dengan nilai bobot matriks *embedding* yang telah di transposisi. Tabel 4.17 menunjukkan nilai bobot matriks *embedding* (*W'*) yang telah di transposisi.

Tabel 4.17 Matriks W'^T

0,8	0,4	0,1
-0,2	0,2	0,6
0,3	0,8	0,1
-0,1	0,1	-0,7
0,7	0,3	0,7
0,9	0,1	0,2
-0,6	0,6	0,5
0,1	0,6	0,2
0,4	0,8	-0,4
0,3	-0,2	0,7
-0,8	0,3	-0,4

Tabel 4.17 Matriks W'^{T} (lanjutan)

0,1	0,2	0,5
0,7	0,1	-0,4
0,3	0,1	-0,7

Berikut contoh perhitungan manual untuk mendapatkan nilai dari *hidden layer* ke *output layer* pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.6.

$$u_j = \begin{bmatrix} 0{,}8 & 0{,}4 & 0{,}1 \\ -0{,}2 & 0{,}2 & 0{,}6 \\ 0{,}3 & 0{,}8 & 0{,}1 \\ -0{,}1 & 0{,}1 & -0{,}7 \\ 0{,}7 & 0{,}3 & 0{,}7 \\ 0{,}9 & 0{,}1 & 0{,}2 \\ -0{,}6 & 0{,}6 & 0{,}5 \\ 0{,}1 & 0{,}6 & 0{,}2 \\ 0{,}4 & 0{,}8 & -0{,}4 \\ 0{,}3 & -0{,}2 & 0{,}7 \\ -0{,}8 & 0{,}3 & -0{,}4 \\ 0{,}1 & 0{,}2 & 0{,}5 \\ 0{,}7 & 0{,}1 & -0{,}4 \\ 0{,}3 & 0{,}1 & -0{,}7 \end{bmatrix} x \begin{bmatrix} 0{,}4 \\ 0{,}1 \\ 0{,}3 \end{bmatrix} = \begin{bmatrix} 0{,}39 \\ 0{,}12 \\ 0{,}23 \\ -0{,}24 \\ 0{,}52 \\ 0{,}43 \\ -0{,}03 \\ 0{,}16 \\ 0{,}12 \\ 0{,}31 \\ -0{,}41 \\ 0{,}21 \\ 0{,}17 \\ -0{,}08 \end{bmatrix}$$

4.2.3.5 *Softmax output*

Dalam melakukan perhitungan *softmax output*, hasil yang didapatkan dari perhitungan *hidden layer* ke *output layer* dapat dihitung dengan menggunakan rumus eksponensial. Berikut contoh perhitungan manual untuk mendapatkan nilai *softmax output* pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.7.

$$y(Paranormal)$$

$$= \frac{\exp(\text{u Paranormal})}{\begin{array}{c}\exp(\text{u Paranormal}) + \exp(\text{u a}) + \exp(\text{u and}) + \exp(\text{u bullies}) + \exp(\text{u clown}) \\ + \exp(\text{u investigator}) + \exp(\text{u malicious}) + \exp(\text{u monster}) + \exp(\text{u of}) \\ + \exp(\text{u shape}) + \exp(\text{u spirits}) + \exp(\text{u takes}) + \exp(\text{u that}) + \exp(\text{u the})\end{array}}$$

$$= \frac{1{,}476}{\begin{array}{c}1{,}476 + 1{,}127 + 1{,}258 + 0{,}786 + 1{,}682 + 1{,}537 + 0{,}971 + 1{,}173 + 1{,}127 + 1{,}363 \\ +0{,}663 + 1{,}233 + 1{,}185 + 0{,}923\end{array}}$$

$$= 0{,}089461836$$

Perhitungan dilakukan pada setiap kata, sehingga menghasilkan nilai *softmax output* seperti yang terdapat pada Tabel 4.18.

Tabel 4.18 *Softmax output*

Kata	*Softmax output*
a	0,089461836
and	0,068293331
bullies	0,076234348
clown	0,047646641
investigator	0,101881678
malicious	0,093112843
monster	0,058780615
of	0,071080435
Paranormal	0,068293331
shape	0,082583683
spirits	0,040197794
takes	0,074724807
that	0,071794805
the	0,055913851

4.2.3.6 Nilai *error*

Dalam melakukan perhitungan nilai *error*, hasil yang didapatkan berasal dari perhitungan nilai *softmax output* dikurangi dengan nilai dari vektor kata konteks (w(c)) dan dilakukan perhitungan nilai rata-rata dari jumlah *error* sebanyak jumlah konteks yang tersedia. Apabila terdapat 2 nilai vektor kata konteks, maka terdapat 2 nilai *error*. Berikut contoh perhitungan manual untuk mendapatkan nilai *error* pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.8.

E(Paranormal) = y(Paranormal) – x(Paranormal di vektor kata konteks 1)

$= 0{,}08946184 - 0$

$= 0{,}08946184$

$$\text{Penjumlahan nilai } error = \frac{\text{nilai } error \text{ konteks } 1 + \text{nilai } error \text{ konteks } 2}{2}$$

$= 0{,}08946184 + 0$

$= 0{,}08946184$

Perhitungan dilakukan untuk setiap kata. Pada kasus dengan kata "Paranormal", vektor kata konteks hanya berjumlah 1 maka hasil yang didapatkan dari rata-rata nilai *error* dapat dilihat pada Tabel 4.19.

Tabel 4.19 Hasil *Error*

Kata	*Error* konteks 1	Penjumlahan nilai *error*
a	0,089461836	0,089462
and	0,068293331	0,068293
bullies	0,076234348	0,076234
clown	0,047646641	0,047647
investigator	-0,89811832	-0,89812
malicious	0,093112843	0,093113
monster	0,058780615	0,058781
of	0,071080435	0,07108
Paranormal	0,068293331	0,068293
shape	0,082583683	0,082584
spirits	0,040197794	0,040198
takes	0,074724807	0,074725
that	0,071794805	0,071795
the	0,055913851	0,055914

4.2.3.7 *Backpropagation*

Untuk meningkatkan nilai bobot, dilakukan proses dengan menggunakan *gradient stochastic descent* (SGD) untuk mengurangi kesalahan yang timbul dengan cara menghitung kesalahan dari *output layer* ke *hidden layer* dan dari *hidden layer* ke *input layer*. Untuk dapat menghitung nilai bobot dari *output layer* ke *hidden layer* dilakukan dengan mengalikan nilai hasil *input layer* ke *hidden layer* (h) dengan nilai *error* yang di transposisi. Tabel 4.20 menunjukkan nilai *error* yang di transposisi.

Tabel 4.20 Hasil *Error* Transposisi

0,0895	0,068	0,0762	0,048	-0,898	0,093	0,0588	0,071	0,0683	0,083	0,0402	0,075	0,0718	0,056

Berikut contoh perhitungan manual untuk mendapatkan nilai dari *output layer* ke *hidden layer* pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.9. Tabel 4.21 menunjukkan hasil nilai nilai dari *output layer* ke *hidden layer* ($\mathrm{dl}'_{\mathrm{dw}}$)

$$\mathrm{dl}'_{\mathrm{dw}} = \begin{bmatrix} 0{,}4 \\ 0{,}1 \\ 0{,}3 \end{bmatrix} \mathrm{x} \begin{bmatrix} 0{,}0895 & 0{,}068 & 0{,}0762 & 0{,}048 & -0{,}0{,}898 & 0{,}093 & 0{,}0588 & 0{,}071 & 0{,}0683 \\ 0{,}083 & 0{,}0402 & 0{,}075 & 0{,}0718 & 0{,}056 & & & & \end{bmatrix}$$

=

$$\begin{bmatrix} 0{,}003 & 0{,}027 & 0{,}030 & 0{,}019 & -0{,}359 & 0{,}037 & 0{,}023 & 0{,}028 & 0{,}027 & 0{,}033 & 0{,}016 & 0{,}03 & 0{,}028 & 0{,}022 \\ 0{,}008 & 0{,}006 & 0{,}007 & 0{,}004 & -0{,}089 & 0{,}009 & 0{,}058 & 0{,}007 & 0{,}007 & 0{,}008 & 0{,}004 & 0{,}007 & 0{,}007 & 0{,}006 \\ 0{,}026 & 0{,}020 & 0{,}022 & 0{,}014 & -0{,}269 & 0{,}027 & 0{,}017 & 0{,}021 & 0{,}0 & 0{,}025 & 0{,}012 & 0{,}022 & 0{,}021 & 0{,}017 \end{bmatrix}$$

Tabel 4.21 Nilai dl'_{dw}

0,035	0,027	0,030	0,019	-0,359	0,037	0,023	0,028	0,027	0,033	0,016	0,03	0,0287	0,022
0,008	0,006	0,007	0,004	-0,089	0,009	0,005	0,007	0,006	0,008	0,004	0,007	0,0072	0,006
0,026	0,020	0,022	0,014	-0,269	0,027	0,017	0,021	0,020	0,025	0,012	0,022	0,0215	0,017

Sedangkan untuk mendapatkan nilai matriks dari perhitungan *hidden layer* ke *input layer*, dilakukan dengan cara mengalikan nilai vektor target (v(t)) dengan hasil perkalian matriks *embedding* (*W'*) x hasil nilai *error* yang di transposisi. Tabel 4.22 menunjukkan nilai *error* yang di transposisi.

Tabel 4.22 Nilai dl'_{dw} transposisi

-0,4026	0,0109	-0,5583

Berikut contoh perhitungan manual untuk mendapatkan nilai dari *hidden layer* ke *input layer* pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.10.

$$\mathrm{dl_{dw}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} x[-0{,}403 \quad 0{,}0109 \quad -0{,}5583] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -0{,}403 & 0{,}0109 & -0{,}558 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Tabel 4.23 menunjukkan hasil nilai nilai dari *hidden layer* ke *input layer* (dl_{dw}).

Tabel 4.23 Nilai dl_{dw}

0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

Tabel 4.23 Nilai dl_{dw} (lanjutan)

0	0	0
0	0	0
-0,4026	0,0109	-0,5583
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

4.2.3.8 Pemutakhiran bobot

Pemutakhiran bobot dilakukan untuk mengubah nilai bobot dari bobot awal matriks konteks (*W*) dan matriks *embedding* (*W'*) menggunakan rumus pada Persamaan 2.11. Nilai yang dihasilkan dari pemutakhiran bobot akan digunakan untuk nilai bobot pada proses berikutnya. Tabel 4.24 menunjukkan hasil pemutakhiran bobot matriks konteks (*W*) dan Tabel 4.25 menunjukkan hasil pemutakhiran bobot matriks *embedding* (*W'*).

Tabel 4.24 Matriks bobot akhir *W*

0,5	0,3	0,5
0,1	-0,6	0,7
0,4	0,8	0,2
-0,6	0,3	-0,1
0,5	0,2	-0,4
0,7	0,6	-0,3
-0,2	0,4	0,1
0,1	0,8	0,2
0,5	0,3	0,5
0,404	0,0999	0,3056
-0,2	0,3	0,6
0,2	0,8	0,1
-0,1	-0,6	0,4
-0,6	0,5	0,2
-0,1	0,3	0,4

Tabel 4.25 Matriks bobot akhir *W'*

0,7996	-0,2003	0,2997	-0,1	0,704	0,8996	-0,6	0,0997	0,4	0,2997	-0,8	0,0997	0,7	0,2998
0,3999	0,19993	0,7999	0,1	0,301	0,0999	0,6	0,5999	0,8	-0,2	0,3	0,1999	0,1	0,0999
0,0997	0,5998	0,0998	-0,7	0,703	0,1997	0,5	0,1998	-0,4	0,6998	-0,4	0,4998	-0,4	-0,7002

Berikut contoh perhitungan manual pemutakhiran bobot pada kata "Paranormal" dengan menggunakan rumus pada Persamaan 2.11.

W(Paranormal) = 0,4-(0,01*-0,403) = 0,404.

4.2.4 Pembuatan *Query Expansion*

Dalam membuat vektor *query* dengan metode *Word2Vec*, dilakukan perhitungan peluang dari *query*. Misalkan kata "clown", merupakan hasil kata tertinggi dari hasil perhitungan TF-IDF, maka dengan metode *Word2Vec* akan dicari kata lain yang memiliki kedekatan dengan kata "clown". Proses dilakukan dengan mencari nilai theta menggunakan rumus pada Persamaan 2.13. Berikut contoh perhitungan manual untuk menghitung nilai theta dari kata *query* "clown" dan kata konteks "a".

$$\theta = \frac{[-0{,}5925 \quad 0{,}30035 \quad -0{,}09936]x[0{,}60598 \quad 0{,}50072 \quad 0{,}20781]^T}{\sqrt{(-0{,}5925)^2 + (0{,}30035)^2 + (-0{,}09936)^2}\ x\ \sqrt{(0{,}60598)^2 + (0{,}50072)^2 + (0{,}20781)^2}} = 0{,}89506$$

Nilai theta yang dihasilkan merupakan perhitungan dari kedekatan antara kata *query* ke semua kata konteks yang ada. Nilai keseluruhan theta diurutkan berdasarkan nilai yang paling tinggi ke yang rendah dan akan diambil 3 kata teratas untuk dijadikan sebagai penambahan *query* pada kata clown, maka penambahan kata untuk kata "clown" yaitu "that" "monster" dan "shape". Tabel 4.26 menunjukkan hasil nilai theta dari seluruh kata yang memiliki kedekatan dengan kata "clown".

Tabel 4.26 Hasil nilai theta

Urutan	Kata
1	clown
0,8950642	that
0,7324573	monster
0,3215558	shape
0,3093944	the
0,2790432	of
0,1922099	spirits
-0,023761	bullies
-0,30572	malicious
-0,33779	takes
-0,430205	investigator
-0,478821	a

Tabel 4.26 Hasil nilai theta (lanjutan)

Urutan	Kata
-0,501296	and
-0,691166	Paranormal

4.2.5 Perhitungan Nilai Akhir

Setelah mendapatkan kata tambahan dari proses *Word2Vec*, maka lakukan penggabungan kata dari hasil TF-IDF dan hasil *Query Expansion*. Dalam menghasilkan proses rekomendasi yang sesuai, lakukan perhitungan nilai kemiripan antar dokumen dengan algoritme *cosine similarity*.

Proses awal untuk menghitung nilai kemiripan antar dokumen dengan algoritme *cosine similarity* yaitu dengan menghitung nilai *Term frequency-invers document frequency* (TF-IDF) dengan penambahan kata yang dihasilkan dari proses *Word2Vec* yaitu "that", "monster" dan "shape". Sebelum dilakukan proses perhitungan, kata yang telah digabungkan akan melalui tahap *stopword removal* atau penghilangan kata yang tidak memiliki arti penting dalam sebuah dokumen. Maka untuk perhitungan manualisasi ini, kata tambahan yang digunakan yaitu "monster" dan "shape". Tabel 4.27 menunjukkan nilai *frequency-invers document frequency* (TF-IDF) yang dihasilkan dari seluruh kata.

Tabel 4.27 *frequency-invers document frequency* ($W_{t,d}$)

TERM	D1	D2	D3	*Query*
paranormal	0,477121	0	0	0
investigators	0,477121	0	0	0
malicious	0	0,477121	0	0
spirits	0	0,477121	0	0
bullies	0	0	0,477121	0
monster	0	0	0,477121	0,477121
takes	0	0	0,477121	0
shape	0	0	0,477121	0,477121
clown	0	0	0,477121	0,477121

Setelah mendapatkan nilai *frequency-invers document frequency* (TF-IDF) yang dihasilkan dari seluruh kata. Lakukan proses normalisasi kata yang ada di dokumen maupun *query*. Berikut contoh perhitungan manual normalisasi data pada kata "clown" yang ada di dalam dokumen menggunakan rumus pada persamaan 2.4.

$$W_{t,d} = \frac{0{,}477121}{\sqrt{0{,}477121^2 + 0{,}477121^2 + 0{,}477121^2}} = \frac{0{,}477121}{0{,}68293} = 0{,}57735$$

Tabel 4.28 menunjukkan hasil normalisasi dari seluruh kata dalam dokumen dan *query*.

Tabel 4.28 Hasil Normalisasi

TERM	D1	D2	D3	*Query*
paranormal	0,707107	0	0	0
investigators	0,707107	0	0	0
malicious	0	0,707107	0	0
spirits	0	0,707107	0	0
bullies	0	0	0,447214	0
monster	0	0	0,447214	0,57735
takes	0	0	0,447214	0
shape	0	0	0,447214	0,57735
clown	0	0	0,447214	0,57735

Untuk mendapatkan hasil kemiripan kata *query* dengan dokumen, lakukan proses perhitungan dengan *cosine similarity* dengan rumus pada Persamaan 2.14. Berikut perhitungan manual untuk menghitung nilai kemiripan antar *query* dengan dokumen.

$$CosSim(D_1, Q) = (0\ x\ 0{,}57735) + (0\ x\ 0{,}57735) + (0\ x\ 0{,}57735) = 0$$

$$\mathrm{CosSim}(\mathrm{D}_2, \mathrm{Q}) = (0\ x\ 0{,}57735) + (0\ x\ 0{,}57735) + (0\ x\ 0{,}57735) = 0$$

$$CosSim(D_3, Q) = (0{,}57735\ x\ 0{,}57735) + (0{,}57735\ x\ 0{,}57735) + (0{,}57735\ x\ 0{,}57735) = 1$$

Setelah mendapatkan hasil berupa nilai kemiripan antara dokumen dengan *query*, proses selanjutnya yaitu mengurutkan nilai dari yang terbesar hingga terkecil. Maka urutan hasil dari proses rekomendasi yang sesuai dengan *query* yaitu **Dokumen 3, Dokumen 1 lalu dokumen 2**.

4.3 Perancangan Pengujian

Proses pengujian dilakukan untuk mengetahui tingkat keberhasilkan algoritme terhadap hasil rekomendasi yang dihasilkan. Terdapat beberapa parameter yang akan dilakukan pengujian yaitu pengujian *Window Size*, *Hidden Neuron*, *Learning Rate*, *Epoch*, *Top-N* kata hasil TF-IDF dan *Top-N* kata hasil *Word2Vec*. Pengujian dilakukan untuk mendapatkan nilai *Precision*, *dan Mean Average Precision* (MAP). Dalam mendapatkan bobot yang optimal pada metode *word2vec*, dilakukan pengujian untuk tiap *hyperparameter* dengan tujuan dapat meminimalkan nilai *error* untuk mencapai nilai yang konvergen. Pengujian dilakukan dengan menggunakan proses *Query Expansion* dan tanpa proses *Query Expansion* untuk dapat membandingkan hasil yang didapatkan.

4.3.1 Perancangan Pengujian *Hidden Neuron*

Perancangan pengujian *hidden neuron* dilakukan untuk mendapatkan nilai terbaik yang dihasilkan oleh metode *Word2vec*. Tabel 4.29 menunjukkan perancangan pengujian *Precision* dan Tabel 4.30 menunjukkan perancangan pengujian *Mean Average Precision* (MAP) untuk *hidden neuron*.

Tabel 4.29 Perancangan Pengujian *Precision@K Hidden Neuron*

Hidden Neuron	Jumlah *Query*	Jumlah Relevan	Jumlah Tiddak Relevan	Rata-rata *Precision*@10
5	30			
10	30			
20	30			
50	30			

Hidden Neuron	*Query*	Relevan	Tidak Relevan	P@10
	Q1			
	Q2			
	.... Q30			
Total/rata-rata				

Tabel 4.30 Perancangan Pengujian MAP untuk *Hidden Neuron*

Jumlah *Hidden Neuron*	Nilai MAP
5	
10	
20	
50	

Hidden Neuron	Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
	Q1											

Tabel 4.30 Perancangan Pengujian MAP untuk *Hidden Neuron* (lanjutan)

	Q2											
	.....											
	Q30											
Nilai MAP												

4.3.2 Perancangan Pengujian *Top-N* kata hasil TF-IDF

Perancangan pengujian *Top-N* kata dilakukan untuk mendapatkan banyaknya kata optimal dari hasil *query* TF-IDF yang akan dilakukan penambahan kata/ *query expansion.* Tabel 4.31 menunjukkan perancangan pengujian *Precision* dan Tabel 4.32 menunjukkan perancangan pengujian *Mean Average Precision* (MAP) untuk mendapatkan *Top-N* kata terbaik dari hasil TF-IDF.

Tabel 4.31 Perancangan Pengujian *Precision Top-N* Kata Hasil TF-IDF

Top-N kata hasil TF-IDF	Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
5				
10				

Top-N	*Query*	Relevan	Tidak Relevan	P@10
	Q1			
	Q2			
	.... Q30			
Total/rata-rata				

Tabel 4.32 Perancangan Pengujian MAP *Top-N* Kata Hasil TF-IDF

Top-N kata hasil TF-IDF	Nilai MAP
5	
10	

Tabel 4.32 Perancangan Pengujian MAP *Top-N* Kata Hasil TF-IDF (lanjutan)

Top-N	Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
	Q1											
	Q2											
	.....											
	Q30											
Nilai MAP												

4.3.3 Perancangan Pengujian *Top-K* hasil *Word2Vec*

Perancangan pengujian *Top-K* kata dilakukan untuk mendapatkan banyaknya kata optimal dari hasil *query expansion* yang akan ditambahkan ke *query* awal. Tabel 4.33 menunjukkan perancangan pengujian *Precision* dan Tabel 4.34 menunjukkan perancangan pengujian *Mean Average Precision* (MAP) untuk mendapatkan *Top-K* kata terbaik dari hasil *query expansion.*

Tabel 4.33 Perancangan Pengujian *Precision Top-K* Kata Hasil *Word2Vec*

Top-K kata hasil *Word2Vec*	Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
5				
10				
50				

Top-K	*Query*	Relevan	Tidak Relevan	P@10
	Q1			
	Q2			
	.... Q30			
Total/rata-rata				

Tabel 4.34 Perancangan Pengujian MAP *Top-K* Kata Hasil *Word2Vec*

Top-K kata hasil *Word2Vec*	Nilai MAP
5	
10	
50	

Top-K	Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
	Q1											
	Q2											
	....											
	Q30											
Nilai MAP												

4.3.4 Perancangan Pengujian Tanpa *Query Expansion*

Perancangan pengujian tanpa *Query Expansion* dilakukan untuk mendapatkan nilai yang akan dibandingkan dengan hasil nilai menggunakan *Query Expansion*. Tabel 4.35 menunjukkan perancangan pengujian *Precision@k* dan Tabel 4.36 menunjukkan perancangan pengujian *Mean Average Precision* (MAP).

Tabel 4.35 Perancangan Pengujian *Precision@k* Tanpa *Query Expansion*

Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
30	136	163	0,45333

Query	Relevan	Tidak Relevan	P@10
Q1			
Q2			
.... Q30			
Total/rata-rata			

Tabel 4.36 Perancangan Pengujian MAP Tanpa *Query Expansion*

Jumlah *Query*	Nilai MAP
30	

Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
Q1											
Q2											
....											
Q30											
Nilai MAP											

4.4 Implementasi

Dalam tahap implementasi, Subbab ini membahas pengimplementasian proses rekomendasi berupa kode program beserta penjelasannya. Terdapat beberapa kode program untuk membuat sebuah proses rekomendasi yaitu kode program proses ambil data, *Pre-processing*, pembuatan nilai bobot dengan TF-IDF, pembuatan matriks bobot dengan *Word2Vec*, pembuatan *Query Expansion* dan perhitungan nilai akhir.

4.4.1 Implementasi Ambil Data

Proses pengambilan data dilakukan dengan mengambil sinopsis sebuah film dari halaman web MovieDb secara otomatis dengan memanggil id dari film tersebut. Proses pengambilan data dari HTML menggunakan *library* beautifulsoup. Kode Program 4.1 merupakan implementasi kode program dari ambil data.

Algoritme 1: Pengambilan data

```
1  dataFilm = []
2  idFilm = []
3  i = 0
4  j = 0
5
6  file = open("dataLatih2.txt", "r")
7  idFilm = file.read().splitlines()
8  for i in range(len(idFilm)):
9      idFilm[i] = idFilm[i].split("-")
10 file.close()
11
12 base_url = 'https://www.themoviedb.org/movie/'
```

```
13 while j < len(idFilm):
14     page = requests.get(base_url + idFilm[j][0])
15     soup = BeautifulSoup(page.content, 'html.parser')
16     overview_box    =    soup.find('div',    attrs={'class':
   'overview'})
17     dataFilm.append([idFilm[j], overview_box.text.strip()])
18     j += 1
19
20 with open('dataLatih2.pkl', 'wb') as f:
21    pickle.dump(dataFilm, f)
22    f.close()
```

Kode Program 4.1 Implementasi proses ambil data

Terdapat penjelasan dari kode program 4.1 mengenai implementasi proses ambil data yaitu:

1. Baris 1-2 merupakan array dataFilm dan idFilm.
2. Baris 6-10 merupakan proses membaca file pickle dan mengambil idFilm yang terpisah dengan tanda (-) kemudian memasukkannya ke dalam array idFilm.
3. Baris 12-18 merupakan proses mengambil data pada halaman web MovieDb berdasarkan idFilm.
4. Baris 20-22 merupakan proses memasukkan data ke dalam pickle.

4.4.2 Implementasi *Pre-Processing*

Proses *pre-processing* merupakan tahapan awal dalam *Text processing* untuk penyaringan data dari kata yang tidak memiliki makna. *Pre-processing* pada penelitian ini meliputi proses tokenisasi, *case folding* dan *stopword removal* menggunakan bantuan *library* NLTK. Kode Program 4.2 merupakan implementasi kode program dari *pre-processing*.

Algoritme 2: Proses *Pre-processing*

```
1  token = []
2  hasilStopword = []
3
4  with open('dataLatih.pkl', 'rb') as f:
5      dataset = pickle.load(f)
6
7  #tokenisasi dan case folding
8  i = 0
9  while i < len(dataset):
10     temp = re.findall(r"[\w']+", dataset[i][1].lower())
11     token.append([dataset[i][0][1], temp])
12     i += 1
13
14 #stopword
15 stop_words = set(stopwords.words('english'))
16 i = 0
17 while i < len(dataset):
18     word_tokens = token[i][1]
19     filtered_sentence = []
20     for w in word_tokens:
21         if w not in stop_words:
22             filtered_sentence.append(w)
```

```
23      hasilStopword.append([dataset[i][0][1],
    filtered_sentence])
24      i += 1
25
26  with open('hasilStopword.pkl', 'wb') as f:
27      pickle.dump(hasilStopword, f)
28      f.close()
29  print("Preprocessing Selesai")
```

Kode Program 4.2 Implementasi proses *Pre-processing*

Terdapat penjelasan dari kode program 4.2 mengenai implementasi proses *pre-processing* yaitu:

1. Baris 1-2 merupakan deklarasi array token[] dan hasilstopword[]
2. Baris 4-5 proses membaca file pickle dan memindahkannya ke dalam array.
3. Baris 8-12 merupakan proses pemotongan kata (tokenisasi), penghapusan karakter selain huruf dan mengubah kata menjadi *lower case*.
4. Baris 15-24 merupakan proses penghilangan *stopword* atau menghilangkan kata yang tidak memiliki makna dalam sebuah dokumen dan memindahkannya ke dalam array *filtered_sentence*.
5. Baris 26-28 merupakan proses memasukkan data ke dalam pickle.

4.4.3 Implementasi Pembuatan Nilai Bobot dengan TF-IDF

Proses pembuatan nilai bobot untuk setiap kata dengan metode TF-IDF memiliki fungsi yaitu agar setiap kata memiliki nilai bobot sesuai frekuensi kemunculkan kata dalam suatu dokumen untuk dapat digunakan pada proses selanjutnya. Kode Program 4.3 merupakan implementasi kode program dari pembuatan nilai bobot dengan TF-IDF terdapat pada.

```
Algoritme 3: Proses pembuatan bobot kata dengan TF-IDF
 1  querySama = []
 2  with open('hasilStopword.pkl', 'rb') as f:
 3      corpus = pickle.load(f)
 4
 5  for i in range(len(corpus)):
 6      temp = ' '.join(corpus[i][1])
 7      corpus[i] = temp
 8
 9  with open('dataLatih.pkl', 'rb') as f:
10      dataFilm = pickle.load(f)
11
12  #mengambil query berdasarkan film pilihan user
13  print("Daftar Film:")
14  for i in range (len(dataFilm)):
15      print(i+1, dataFilm[i][0][1], sep=". ")
16  filmDipilih = int(input("Pilih Nomor Film: "))
17
18  if(filmDipilih > 200 or filmDipilih < 1):
19      import sys
20      sys.exit("--ERROR! Nomor Film Diluar Dari Daftar Film--")
21  else:
22      query = [dataFilm[filmDipilih-1][1]]
```

```
23      with open('query.pkl', 'wb') as f:
24          pickle.dump(dataFilm[filmDipilih-1], f)
25
26  #proses tf idf
27  vect = TfidfVectorizer(stop_words=None)
28  trainVect = vect.fit_transform(corpus)
29
30  tfidfCorpus = pd.DataFrame(trainVect.toarray(), columns =
    vect.get_feature_names())
31  tfidfWord = vect.get_feature_names()
32
33  x = vect.transform(query)
34  tfidfQuery = vect.transform(query).toarray()
35
36  for i in range(len(tfidfQuery[0])):
37      if(tfidfQuery[0][i] != 0 ):
38          querySama.append([tfidfWord[i], tfidfQuery[0][i]])
39
40  querySama = sorted(querySama,key=lambda l:l[1], reverse=True)
41
42  with open('tfidf_querySama.pkl', 'wb') as f:
43      pickle.dump(querySama, f)
44  print("TFIDF Selesai")
```

Kode Program 4.3 Implementasi Pembuatan Nilai Bobot dengan TF-IDF

1. Baris 2-10 merupakan proses inisialisasi list corpus dan dataFilm dengan data dari pickle.
2. Baris 13-16 merupakan proses menampilkan film untuk selanjutnya dipilih oleh *user.*
3. Baris 18-20 merupakan proses menampilkan pesan *error* apabila masukan melebihi pilihan yang ada.
4. Baris 22-24 merupakan proses memasukkan data ke dalam pickle.
5. Baris 27-34 merupakan proses TF-IDF dengan bantuan *library Tfidfvectorizer*.
6. Baris 36-40 merupakan proses inisialisasi *list* querySama dan pengurutannya berdasarkan nilai TF-IDF.
7. Baris 42-43 merupakan proses memasukan data ke dalam pickle untuk *list* querySama.

4.4.4 Implementasi Pembuatan Matriks Bobot dengan *Word2Vec*

Proses pembuatan matrix bobot dengan metode *Word2Vec* akan membentuk sebuah vektor kata, dimana kata tersebut memiliki hubungan semantik maupun sintaksis. Proses yang terdapat pada *word2vec* yaitu menentukan bobot kata target dan konteks, menghitung nilai *hidden layer*, menghitung nilai *output layer*, menghitung nilai *softmax output*, menghitung nilai *error*, proses *backpropagation* serta menghitung pemutakhiran bobot. Kode Program 4.4 merupakan implementasi kode program dari pembuatan *matrix* bobot dengan *word2vec*.

Algoritme 4: Proses pembuatan bobot dengan metode *word2vec*

```
#inisialisasi
kalimatPerdokumen = []

with open('dataLatih.pkl', 'rb') as f:
    dataLatih = pickle.load(f)

for i in range(len(dataLatih)):
    temp = dataLatih[i][1]
    kalimatPerdokumen.append(temp)

hidden_neuron = int(input("Hidden Neuron: "))
window_size = int(input("Window Size: "))
learning_rate = 0.03
epoch = 1
token = []
vocab = []
wt = []
wc = []

#potongkata
for i in range(len(kalimatPerdokumen)):
    kalimat = kalimatPerdokumen[i].split(".")
    for j in range(len(kalimat)):
        temp = re.findall(r"[\w']+", kalimat[j])
        token.append([i, j, temp])
tempToken = []
for i in range(len(token)):
    if token[i][2] != []:
        tempToken.append(token[i])
token = tempToken

#inisialisasi vektor kata target dan konteks
for i in range(len(token)):
    vocab += token[i][2]

#sort and remove duplicate w_t
vocab = np.unique(vocab).tolist()
vocab.sort(key=str.lower)
i = 0
while i < len(token):
    j = 0
    while j < len(token[i][2]):
        temp = []
        for x in vocab:
            if x == token[i][2][j]:
                temp += [1]
            else:
                temp += [0]
        wt.append([i , token[i][2][j], temp])
        j += 1
    i += 1

i = 0
ik = 0
while i < len(token):
    j = 0
    while j < len(token[i][2]):
       wc.append([token[i][0],token[i][1],token[i][2][j]])
```

```
59          k = 1
60          while k <= window_size:
61              wcSebelum = []
62              wcSesudah = []
63              if j - k < 0:
64                  for x in vocab:
65                      wcSebelum += [0]
66              else:
67                  for x in vocab:
68                      if x == token[i][2][j-k]:
69                          wcSebelum += [1]
70                      else:
71                          wcSebelum += [0]
72              if j + k >= len(token[i][2]):
73                  for x in vocab:
74                      wcSesudah += [0]
75              else:
76                  for x in vocab:
77                      if x == token[i][2][j+k]:
78                          wcSesudah += [1]
79                      else:
80                          wcSesudah += [0]
81              wc[ik] += [wcSebelum,wcSesudah]
82              k += 1
83          j += 1
84          ik += 1
85      i += 1
86
87  #INISIALISASI NILAI BOBOT AWAL
88  n = len (vocab)
89  w1_konteks_random = np.random.randn(n, hidden_neuron) #w1
90  w2_embedding_random = np.random.randn(hidden_neuron, n) #w2
91
92  #pembulatan nilai
93  w1_konteks = np.round(w1_konteks_random,1) #w
94  w2_embedding = np.round(w2_embedding_random,1) #w'
95
96  for a in range (epoch):
97      for indeksKata in range(len(wt)):
98          #forward pass
99          w1_t = np.transpose(w1_konteks)
100         w2_t = np.transpose(w2_embedding)
101         h = np.zeros([len(w1_t),1])
102         wtNow=np.reshape(wt[indeksKata][2],
    (len(wt[indeksKata][2]),1))
103         h = np.dot(w1_t, wtNow)
104         u = np.zeros([len(w2_t), 1])
105         u = np.dot(w2_t, h)
106
107         #softmax output
108         yc = []
109         sumExp = sum(np.exp(u))
110         for i in range(len(u)):
111             expNow = np.exp(u[i])
112             temp = expNow/sumExp
113             yc.append(temp)
114
115         #hitung nilai error
116         e = np.zeros((len(yc), 1))
```

```
117         i = 3
118         while i < len(wc[indeksKata]):
119             tempError = []
120             if(sum(wc[indeksKata][i]) == 0):
121                 tempError= np.zeros(len(wc[indeksKata][i]))
122                 tempError=np.reshape(tempError,
    (len(tempError),1))
123             else:
124                 for j in range(len(wc[indeksKata][i])):
125                     temp = yc[j] - wc[indeksKata][i][j]
126                     tempError += [temp]
127             e = np.add(e, tempError)
128             i += 1
129
130         #backpropagation
131         eTranspose = np.transpose(e)
132         dl_dw2 = np.dot(h,eTranspose)
133         w2_embeddingMatrix = np.matrix(w2_embedding)
134         x = np.dot(w2_embeddingMatrix,e)
135         wtNow=np.reshape(wt[indeksKata][2],
    (len(wt[indeksKata][2]),1))
136         xTranspose = np.transpose(x)
137         dl_dw1 = np.dot(wtNow,xTranspose)
138
139         #update bobot
140         w1_konteksNow=w1_konteks-
    (np.dot(learning_rate,dl_dw1))
141         w2_embeddingNow=w2_embedding-(np.dot(learning_rate,
    dl_dw2))
142         w1_konteksNow=np.array(w1_konteksNow)
143         w2_embeddingNow=np.array(w2_embeddingNow)
144         w1_konteks = w1_konteksNow
145         w2_embedding = w2_embeddingNow
146
147 with open('w1_konteks.pkl', 'wb') as f:
148     pickle.dump(w1_konteks, f)
149
150 with open('vocab.pkl', 'wb') as f:
151     pickle.dump(vocab, f)
152 print ("Proses Word2Vec Selesai")
```

Kode Program 4.4 Implementasi Pembuatan Matriks Bobot dengan *Word2Vec*

Terdapat penjelasan dari kode program 4.4 mengenai implementasi proses pembuatan matriks bobot menggunakan *word2vec* yaitu:

1. Baris 4-9 merupakan proses inisialisasi *list* kalimatPerdokumen dengan data dari pickle.
2. Baris 11-14 merupakan proses inisialisasi nilai *hidden neuron*, jumlah *window size*, nilai *learning rate*, dan jumlah *epoch*.
3. Baris 15-18 merupakan deklarasi array.
4. Baris 21-30 merupakan proses memotong kalimat menjadi kata per kata.
5. Baris 33-51 merupakan proses membuat bobot vektor kata target.

6. Baris 53-85 merupakan proses membuat bobot vektor kata konteks sesuai dengan jumlah *window size*.
7. Baris 88-94 merupakan proses Inisialisasi nilai bobot awal secara acak.
8. Baris 99-105 merupakan proses menghitung nilai *forward pass* yang terdiri dari proses perhitungan *hidden layer* (h), dan *output layer* (u).
9. Baris 108-113 merupakan proses menghitung nilai *softmax output.*
10. Baris 116-128 merupakan proses menghitung nilai *error.*
11. Baris 131-137 merupakan proses *backpropagation*.
12. Baris 140-145 merupakan proses pemutakhiran bobot.
13. Baris 147-151 merupakan proses memasukan hasil nilai bobot dari *Word2vec* kedalam pickle.

4.4.5 Implementasi Pembuatan *Query Expansion*

Proses pembuatan *query expansion* merupakan proses yang bertujuan untuk mencari jarak antar kata dan mendapatkan kata tambahan yang merepresentasikan sebuah informasi ke dalam *query* yang sesuai sehingga dapat memperkuat hasil rekomendasi. Proses dari *query expansion* menggunakan nilai matriks bobot hasil dari *word2vec* untuk mendapatkan nilai jarak antar kata dengan menghitung nilai num, denum dan tetha. Kode Program 4.5 merupakan implementasi kode program dari pembuatan *query expansion*.

Algoritme 5: Proses pembuatan *query expansion*

```
1  theta = []
2  v_w1 = []
3  query = []
4
5  with open('w1_konteks_WS30.pkl', 'rb') as f:
6      wKonteks = pickle.load(f)
7  with open('vocab.pkl', 'rb') as f:
8      vocab = pickle.load(f)
9  with open('tfidf_querySama.pkl', 'rb') as f:
10     queryDariTFIDF = pickle.load(f)
11
12 topN = int(input("Top N: "))
13 if(topN > len(queryDariTFIDF)):
14     import sys
15     sys.exit("--ERROR! Kata yang diambil melebihi jumlah
   query yang ada--")
16 else:
17     for i in range(topN):
18         query.append(queryDariTFIDF[i])
19
20 #mencari nilai v_w1 untuk setiap query
21 for i in range(len(query)):
22     for j in range(len(vocab)):
23         if(query[i][0] == vocab[j].lower()):
24             v_w1.append(wKonteks[j])
25
26 #mencari nilai theta
```

```
for i in range(len(query)):
    v_w1Now = np.reshape(v_w1[i], (1,len(v_w1[i])))
    for j in range(len(wKonteks)):
        v_w2 = wKonteks[j]
        v_w2Matrix = np.matrix(v_w2)
        v_w2Transpose = np.transpose(v_w2Matrix)
        num = np.dot(v_w1Now, v_w2Transpose)

        #cari denum
        jumlahKuadratV_w1 = 0
        for k in range(len(v_w1Now[0])):
            temp = v_w1Now[0][k]*v_w1Now[0][k]
            jumlahKuadratV_w1 += temp
        akarJumlahKuadratV_w1 = math.sqrt(jumlahKuadratV_w1)

        jumlahKuadratV_w2 = 0
        for l in range(len(v_w2)):
            temp = v_w2[l]*v_w2[l]
            jumlahKuadratV_w2 += temp
        akarJumlahKuadratV_w2 = math.sqrt(jumlahKuadratV_w2)

        denum = akarJumlahKuadratV_w1*akarJumlahKuadratV_w2

        #cari theta
        temp = num/denum
        temp = np.asscalar(temp)
        theta.append([query[i][0],vocab[j], temp])

theta = sorted(theta,key=lambda l:(l[0],l[2]), reverse=True)

#hapus kata yang bernilai 1 atau sama dengan pembanding
for i in range(len(query)):
    for j in thetaTop[:]:
        if(query[i][0] == j[1].lower()):
            thetaTop.remove(j)

dataframe = pd.DataFrame(theta)

topK = int(input("Top K: "))
thetaTop = dataframe.groupby(0).head(topK+1)
thetaTop = thetaTop.values.tolist()

with open('penambahanQuery.pkl', 'wb') as f:
    pickle.dump(thetaTop, f)
print("Query Expansion Selesai")
```

Kode Program 4.5 Implementasi Pembuatan *Query Expansion*

Terdapat penjelasan dari kode program 4.5 mengenai implementasi proses *query expansion* yaitu:

1. Baris 1-3 merupakan proses deklarasi array.
2. Baris 5-10 merupakan proses inisialisasi variabel dari data pickle.
3. Baris 12-18 merupakan proses menyiapkan kata dari TF-IDF sebanyak jumlah yang diinginkan *user.*

4. Baris 21-24 merupakan proses memberikan nilai W pada kata yang berasal dari TF-IDF.
5. Baris 27-53 merupakan proses mencari nilai theta untuk setiap kata dengan proses perhitungan num dan denum terlebih dahulu.
6. Baris 28-33 merupakan proses menghitung nilai num
7. Baris 36-40 merupakan proses menghitung nilai denum
8. Baris 51-53 merupakan proses menghitung nilai theta
9. Baris 55 merupakan proses membuat *list* theta yang telah diurutkan sebanyak jumlah yang diinginkan *user.*
10. Baris 58-61 merupakan proses menghapus *list* yang berisi data ganda
11. Baris 69-70 merupakan proses menyimpan data *query expansion* ke dalam pickle.

4.4.6 Implementasi Perhitungan Nilai Akhir

Setelah mendapatkan kata tambahan dari proses *Word2Vec*, maka lakukan penggabungan kata dari hasil TF-IDF dan hasil *Query Expansion*. Dalam menghasilkan proses rekomendasi yang sesuai, lakukan perhitungan nilai kemiripan antar dokumen dengan algoritme *cosine similarity.* Proses awal untuk menghitung nilai kemiripan antar dokumen dengan algoritme *cosine similarity* yaitu dengan menghitung nilai TF-IDF menggunakan bantuan *library sklearn*. Kode Program 4.6 merupakan implementasi kode program dari perhitungan nilai akhir.

Algoritme 6: Proses perhitungan nilai akhir

```
1  file = open("datauji.txt", "r")
2  idFilm = file.read().splitlines()
3  dataUji = []
4  for i in range(len(idFilm)):
5      if(idFilm[i][-1:] == '0'):
6          dataUji.append([idFilm[i].split("-"),[]])
7          continue
8      else:
9          k = len(dataUji)-1
10         dataUji[k][1] += [idFilm[i].split("-")]
11
12 querySama = []
13 strPenambahanQuery = ''
14 with open('penambahanQuery.pkl', 'rb') as f:
15     penambahanQuery = pickle.load(f)
16 with open('hasilStopword.pkl', 'rb') as f:
17     corpus = pickle.load(f)
18
19 for i in range(len(corpus)):
20     temp = ' '.join(corpus[i][1])
21     corpus[i] = temp
22
23 #ambil data film
24 with open('dataLatih.pkl', 'rb') as f:
25     dataFilm = pickle.load(f)
26 print("pilih Film Yang Sama Dengan Proses TFIDF")
```

```
for i in range(len(dataUji)):
    print(str(i+1) + ". " + str(dataUji[i][0][1]))
filmDiuji = int(input("Film Yang Akan Diuji: "))
filmDipilih = []
for i in range(len(dataFilm)):
    if(dataUji[filmDiuji-1][0][0] == dataFilm[i][0][0]):
        queryAwal = dataFilm[i][1]
        filmDipilih = dataFilm[i]

for i in range(len(penambahanQuery)):
    strPenambahanQuery += " " + penambahanQuery[i][1]

penambahanQuery = [strPenambahanQuery]

queryAkhir = [queryAwal + penambahanQuery[0]]

#tf-idf
vect = TfidfVectorizer(stop_words=None)
trainVect = vect.fit_transform(corpus)

tfidfCorpus = pd.DataFrame(trainVect.toarray(), columns =
vect.get_feature_names())
tfidfWord = vect.get_feature_names()

x = vect.transform(queryAkhir)
tfidfQuery = vect.transform(queryAkhir).toarray()

for i in range(len(tfidfQuery[0])):
    if(tfidfQuery[0][i] != 0 ):
        querySama.append([tfidfWord[i], tfidfQuery[0][i]])

querySama = sorted(querySama,key=lambda l:l[1], reverse=True)

#menghitung cosine similarity
a = cosine_similarity(trainVect, x)

filmMirip = []
for i in range(len(a)):
    if(a[i][0] != 0):
        filmMirip.append([i, a[i][0]])

filmMirip = sorted(filmMirip, key=lambda x: -x[1])
print("10 Film teratas yang mirip dengan " +
filmDipilih[0][1])
filmDitampilkan = 10
top10 = []
for i in range(len(filmMirip)):
    if(dataFilm[filmMirip[i][0]][0][0]== filmDipilih[0][0]):
        continue
    else:
        print(str(i)+"."+
str(dataFilm[filmMirip[i][0]][0][1]))
        top10.append(dataFilm[filmMirip[i][0]][0])
    if(i == 10):
        break
```

Kode Program 4.6 Implementasi Perhitungan Nilai Akhir

Terdapat penjelasan dari kode program 4.6 mengenai implementasi perhitungan nilai akhir yaitu:

1. Baris 1-10 merupakan proses menampilkan data uji dari file .txt.
2. Baris 14-21 merupakan proses deklarasi dan inisialisasi variabel dengan data dari pickle.
3. Baris 24-34 merupakan proses mengambil *query* awal sesuai dengan pilihan *user*.
4. Baris 36-41 merupakan proses menggabungkan kata dari proses *query expansion* ke dalam *query* awal.
5. Baris 44-51 merupakan proses menghitung TF-IDF.
6. Baris 53-57 merupakan proses pengecekan dan pengurutan nilai *query* berdasarkan nilai paling tinggi.
7. Baris 60 merupakan proses menghitung *Cosine Similarity.*
8. Baris 62-67 merupakan proses inisialisasi *list* berisi rekomendasi kemiripan film berdasarkan hasil *cosine similarity.*
9. Baris 68-78 merupakan proses mencetak hasil rekomendasi berdasarkan jumlah yang diinginkan *user.*

4.4.7 Implementasi Proses Evaluasi

Dalam mengevaluasi hasil rekomendasi, dilakukan perhitungan nilai menggunakan *Precision@k* dan *Mean Average Precision* @k (MAP) dimana k merupakan hasil rekomendasi sebanyak *Top-N* yang akan dicocokkan dengan data penguji. Proses evaluasi digunakan untuk menghitung nilai keakuratan sebuah metode dalam menyelesaikan sebuah permasalahan. Kode Program 4.7 merupakan implementasi kode program dari proses evaluasi.

Algoritme 7: Proses evaluasi

```
1  tagRelevant = str(input("Masukan Tag Relevant: "))
2  tagRelevant = tagRelevant.split(",")
3
4  #Mean Average Precision
5  r = 0
6  jumlahR = 0
7  jumlahNR = 0
8  Rlist = []
9  for i in range(len(tagRelevant)):
10     if(tagRelevant[i] == "R"):
11         r += 1
12         jumlahR += 1
13         temp = r/(i+1)
14         Rlist.append(temp)
15     else:
16         temp = 0
17         jumlahNR += 1
18         Rlist.append(temp)
19     print(tagRelevant[i] + " - " + str(temp))
20
```

```
21 if sum(Rlist) == 0:
22     MAP = 0
23     print("MAP = 0")
24 else:
25     MAP = sum(Rlist)/r
26     print("MAP = " + str(MAP))
27 
28 #rata2precision
29 if sum(Rlist) == 0:
30     precision = 0
31     print("Precision@10 = 0")
32 else:
33     precision = r/filmDitampilkan
34     print("Jumlah Relevan = "+str(r))
35     print("Precision@10 = "+str(precision))
36 
37 #simpan ke pickle
38 try:
39     foo = pickle.load(open("hasilPengujian.pkl", "rb"))
40     temp                =                [dataUji[filmDiuji-
   1][0],precision,MAP,jumlahR,jumlahNR]
41     foo.append(temp)
42     with open('hasilPengujian.pkl', 'wb') as f:
43         pickle.dump(foo, f)
44 except (OSError, IOError):
45     temp                =                [[dataUji[filmDiuji-
   1][0],precision,MAP,jumlahR,jumlahNR]]
46     with open('hasilPengujian.pkl', 'wb') as f:
47         pickle.dump(temp, f)
48 
49 #tampilkan seluruh hasil pengujian
50 foo = pickle.load(open("hasilPengujian.pkl", "rb"))
51 rTotal = 0
52 MAPTotal = 0
53 jumlahR = 0
54 jumlahNR = 0
55 for i in range(len(foo)):
56     rTotal += foo[i][1]
57     MAPTotal += foo[i][2]
58     jumlahR += foo[i][3]
59     jumlahNR += foo[i][4]
60 
61 rTotal = rTotal/len(foo)
62 MAPTotal = MAPTotal/len(foo)
63 
64 print("Jumlah film yang telah diuji : " + str(len(foo)))
65 print("Rata-rata Precision@K : " + str(rTotal))
66 print("Mean Average Precision : " + str(MAPTotal))
67 print("Jumlah Relevant : " + str(jumlahR))
68 print("Jumlah Not Relevant : " + str(jumlahNR))
```

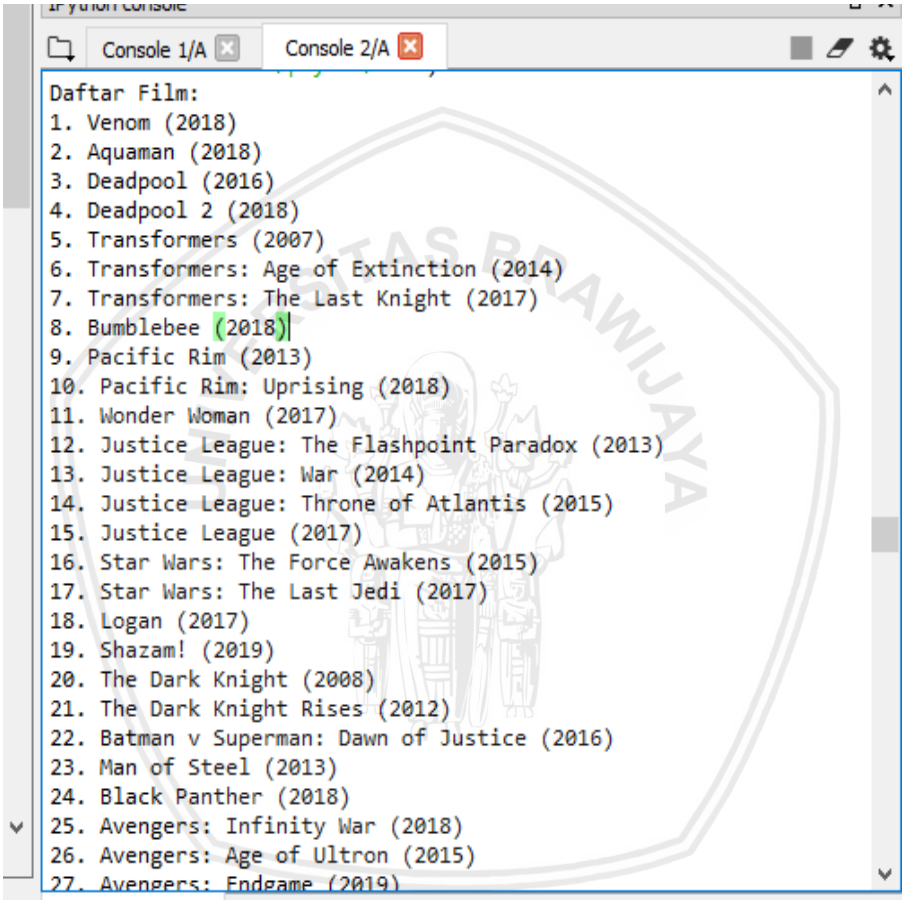
Kode Program 4.7 Implementasi proses evaluasi

Terdapat penjelasan dari kode program 4.7 mengenai implementasi proses evaluasi data yaitu:

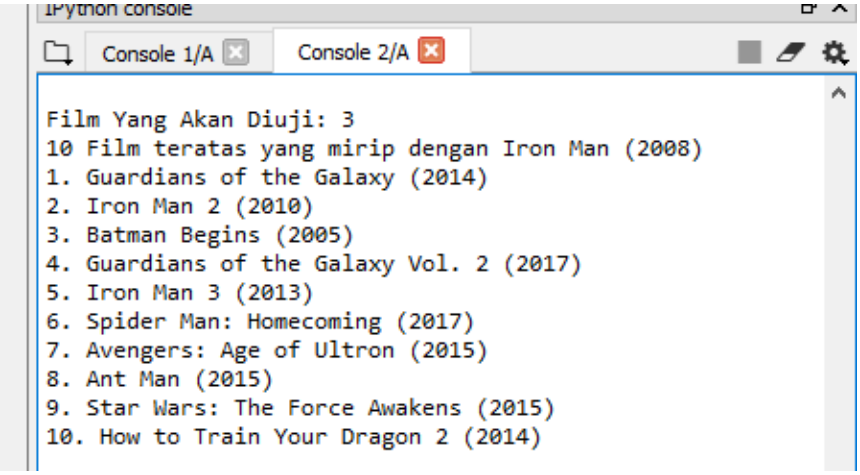
1. Baris 1-2 merupakan proses menerima *input tag* relevan atau tidak relevan.

2. Baris 5-26 merupakan proses pencarian nilai *Mean Average Precision* berdasarkan *tag* relevan atau tidak relevan yang diberikan *user.*
3. Baris 29-35 merupakan proses pencarian nilai rata-rata *Precision@K* berdasarkan *tag* relevan *atau* tidak relevan.
4. Baris 38-47 merupakan proses penyimpanan hasil evaluasi ke dalam pickle.
5. Baris 50-68 merupakan proses mencetak hasil evaluasi.

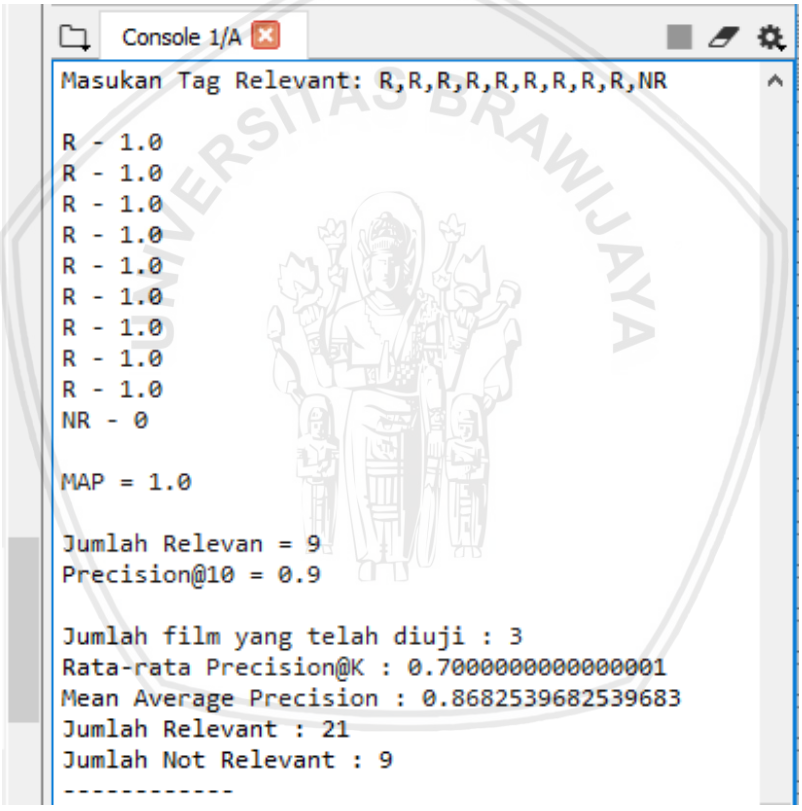
4.5 Tampilan Program



Gambar 4.9 Tampilan Program Daftar Pilihan Film



Gambar 4.10 Hasil Rekomendasi dari Film Iron Man



Gambar 4.11 Hasil Evaluasi dari Film Iron Man

BAB 5
PENGUJIAN DAN ANALISIS

Proses rekomendasi film menggunakan 150 data latih berupa judul beserta sinopsis film berbahasa inggris yang diambil melalui halaman web MovieDb. Proses pengujian menggunakan 30 data uji berupa judul beserta sinopsis film yang dipilih oleh penguji sesuai dengan list judul film yang ada pada data latih. Parameter yang digunakan untuk menguji keakuratan metode *Word2Vec* terdiri dari enam yaitu *learning rate*, *epoch*, *window size*, *hidden neuron*, *Top-N* kata hasil TF-IDF dan *Top-K* kata hasil *Word2Vec*. Nilai parameter yang digunakan merujuk pada penelitian (Caselles-Dupré, Lesaint dan Royo-Letelier, 2018) mengenai *hyperparameter* optimal untuk proses rekomendasi dan merujuk pada parameter *default Word2Vec* dari gensim.

5.1 Pengujian dan Analisis Pengaruh Nilai *Hidden Neuron*

Inisialiasai nilai awal parameter pada pengujian terhadap jumlah *hidden neuron* menggunakan parameter nilai *learning rate* sebesar 0,025, jumlah *epoch* sebanyak 110, jumlah *window size* sebanyak 3, *Top-N* kata *query* dari TF-IDF yang mendapat penambahan *query* sebanyak 5, dan *Top-K* kata hasil *query expansion* yang akan ditambah ke *query* awal sebanyak 5. Tabel 5.1 merupakan hasil pengujian *Precision@k* dan Tabel 5.2 merupakan hasil pengujian *Mean Average Precision* (MAP) untuk parameter *Hidden Neuron*.

Tabel 5.1 Hasil Pengujian *Precision@k* untuk *Hidden Neuron*

Hidden Neuron	Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
5	30	138	162	0,46
10	30	141	159	0,47
20	30	127	173	0,42333
50	30	123	177	0,41

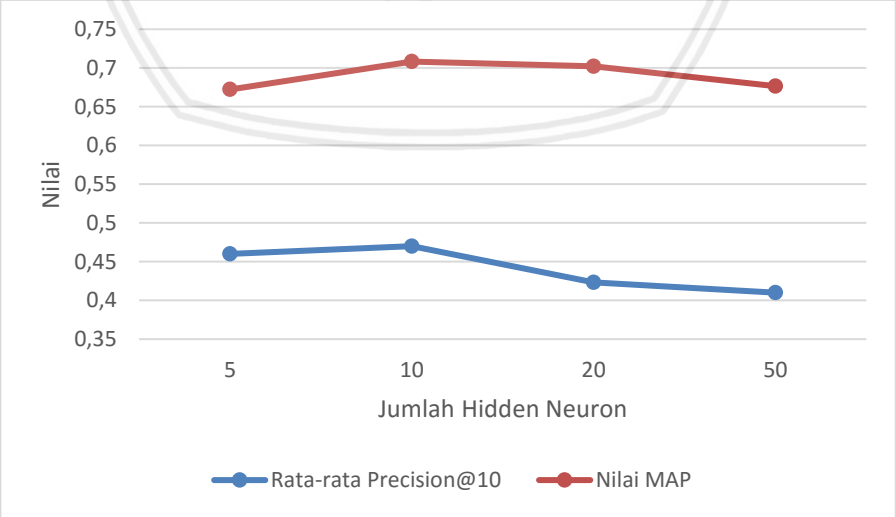
Pengujian *hidden neuron* berdasarkan Tabel 5.1 menunjukkan hasil rata-rata nilai *precision* dari 30 *query* judul film yang masing-masing *query* menghasilkan 10 rekomendasi film. Nilai rata-rata *precision* dari hasil rekomendasi, menunjukkan data relevan yang sesuai dengan *query* meningkat sampai *hidden neuron* 10 yaitu sebesar 0,47, namun pada saat *hidden neuron* lebih dari 10 hasil rekomendasi menunjukkan lebih banyak data yang tidak relevan sehingga nilai rata-rata *Precision* menurun. Nilai rata-rata *precision* yang dihasilkan cenderung tidak mengalami perbedaan yang signifikan dan tergolong rendah, hal tersebut menunjukkan bahwa jumlah hasil rekomendasi yang tidak relevan lebih banyak dibandingkan yang relevan, serta hasil kata dari proses *query expansion* menggunakan metode *Word2Vec* yang ditambahkan dengan *query* awal, banyak yang tidak memiliki kedekatan hubungan makna semantik maupun semantik.

Pengujian pada empat kali percobaan *hidden neuron* menunjukkan nilai rata-rata *Precision* terbesar terdapat pada saat *hidden neuron* 10 sebesar 0,47. Hasil lebih lengkap dari pengujian *Precision@k* untuk *Hidden Neuron* terdapat pada Lampiran 1.

Tabel 5.2 Hasil Pengujian MAP untuk *Hidden Neuron*

Jumlah *Hidden Neuron*	Nilai MAP
5	0,6721888017
10	0,7082408299
20	0,7021896678
50	0,6764684534

Pengujian *hidden neuron* berdasarkan Tabel 5.2 menunjukkan bahwa nilai *Mean Average Precision* (MAP) yang dihasilkan dari 30 *query* proses rekomendasi dengan menampilkan 10 hasil rekomendasi untuk setiap *query* menunjukkan peningkatan pada saat *hidden neuron* bernilai 10 yaitu sebesar 0,70824. Hal tersebut menunjukkan bahwa rata-rata program berhasil menampilkan hasil rekomendasi dengan data yang relevan sesuai dengan peringkat kemiripan data film, namun pada saat *hidden neuron* lebih dari 10 nilai MAP cenderung menurun karena banyak hasil film relevan tidak sesuai dengan peringkat dan hasil kata dari proses *query expansion* menggunakan metode *Word2Vec* yang ditambahkan dengan *query* awal banyak yang tidak memiliki kedekatan hubungan makna sesuai peringkat dokumen. Pengujian pada empat kali percobaan *hidden neuron* menunjukkan nilai *Mean Average Precision* (MAP) terbesar terdapat pada saat *hidden neuron* 10 sebesar 0,7082408299. Hasil lebih lengkap dari pengujian MAP untuk *Hidden Neuron* terdapat pada Lampiran 2.



Gambar 5.1 Grafik Pengujian *Hidden Neuron*

5.2 Pengujian dan Analisis Pengaruh Nilai *Top-N* Kata TF-IDF

Inisialisasi nilai awal parameter pada pengujian terhadap nilai *Top-N* kata dari TF-IDF menggunakan parameter nilai *hidden neuron* sebanyak 10, *learning rate* sebesar 0,025, jumlah *epoch* sebanyak 110, jumlah *window size* sebanyak 3 dan *Top-K* kata hasil *query expansion* yang akan ditambah ke *query* awal sebanyak 5. Tabel 5.3 merupakan hasil pengujian *Precision@k* dan Tabel 5.4 merupakan hasil pengujian *Mean Average Precision* (MAP) *Top-N* kata hasil TF-IDF.

Tabel 5.3 Pengujian *Precision@k Top-N* Kata Hasil TF-IDF

Top-N kata hasil TF-IDF	Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
5	30	141	159	0,47000
10	30	129	171	0,43000

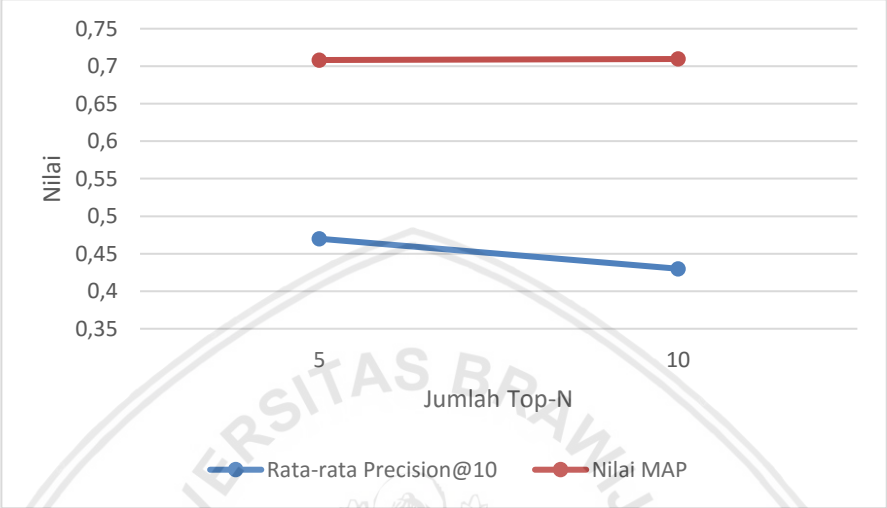
Pengujian *Top-N* kata dari hasil TF-IDF berdasarkan Tabel 5.3 menunjukkan hasil rata-rata nilai *precision* dari 30 *query* judul film yang masing-masing *query* menghasilkan 10 rekomendasi film. Nilai rata-rata *precision* dari hasil rekomendasi menunjukkan data relevan yang sesuai dengan *query* meningkat sampai *Top-N* kata hasil TF-IDF yang diambil sebanyak 5 kata, namun pada saat *Top-N* kata yang diambil lebih dari 5 maka menghasilkan penurunan nilai rata-rata *precision*. Hal tersebut menunjukkan bahwa semakin banyak kata yang dilakukan *query expansion,* maka kata akan semakin bervariasi sehingga tidak cocok untuk studi kasus yang hanya menggunakan 150 data film, karena mengakhibatkan tingkat kemiripan kata akan semakin jauh dengan *query*. Pengujian pada 2 kali percobaan *Top-N* kata hasil TF-IDF menghasilkan nilai rata-rata *precision* terbesar yaitu 0,47 untuk pengambilan kata sebanyak 5. Hasil lebih lengkap dari pengujian *Precision@K Top-N kata* dari TF-IDF terdapat pada Lampiran 3.

Tabel 5.4 Pengujian MAP *Top-N* Kata Hasil TF-IDF

Top-N kata hasil TF-IDF	Nilai MAP
5	0,7082408299
10	0,709603374

Pengujian *Top-N* kata hasil TF-IDF berdasarkan Tabel 5.4 menunjukkan bahwa nilai *Mean Average Precision* (MAP) yang dihasilkan dari 30 *query* proses rekomendasi dengan menampilkan 10 hasil rekomendasi untuk setiap *query,* menunjukkan nilai *MAP* terbesar pada Top-N kata diambil sebanyak 10 dibandingkan 5 meskipun tidak signifikan, padahal nilai rata-rata *Precision*@10 lebih besar dihasilkan pada saat pengambilan kata sebanyak 5 dibandingkan 10. Hal tersebut menunjukkan bahwa rata-rata program berhasil menampilkan hasil rekomendasi dengan data yang relevan sesuai dengan peringkat kemiripan data film lebih banyak pada saat Top-N kata yang diambil sebanyak 10, meskipun data relevan yang dihasilkan lebih sedikit dibandingkan pada saat pengambilan Top-N

kata sebanyak 5. Pengujian MAP dilakukan hanya 2 kali dikarenakan jumlah kata yang sama antara *query* dan data latih terdapat *query* yang tidak lebih dari 10 kata. Dalam 2 kali percobaan, nilai rata-rata *Mean Average Precision* (MAP) terbesar yang dihasilkan yaitu 0,70960 pada saat *Top-N* kata yang akan dilakukan *Query Expansion* sebanyak 10. Hasil lebih lengkap dari pengujian MAP *Top-N kata* dari TF-IDF terdapat pada Lampiran 4.



Gambar 5.2 Grafik Pengujian *Top-N* kata dari TF-IDF

5.3 Pengujian dan Analisis Pengaruh Nilai *Top-K* Kata *Word2Vec*

Inisialiasai nilai awal parameter pada pengujian terhadap nilai *Top-K* kata dari *Word2Vec* menggunakan parameter nilai *hidden neuron* sebanyak 5, *learning rate* sebesar 0,025, jumlah *epoch* sebanyak 5, jumlah *window size* sebanyak 3 dan *Top-N* kata *query* dari TF-IDF yang mendapat penambahan *query* sebanyak 5. Tabel 5.5 merupakan hasil pengujian *Precision@k* dan Tabel 5.6 merupakan hasil pengujian *Mean Average Precision* (MAP) pada Top-K kata hasil *Word2Vec.*

Tabel 5.5 Pengujian *Precision@k Top-K* Kata Hasil *Word2Vec*

Top-K kata hasil *Word2Vec*	Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
5	30	141	159	0,47000
10	30	135	165	0,44999
50	30	123	177	0,41000

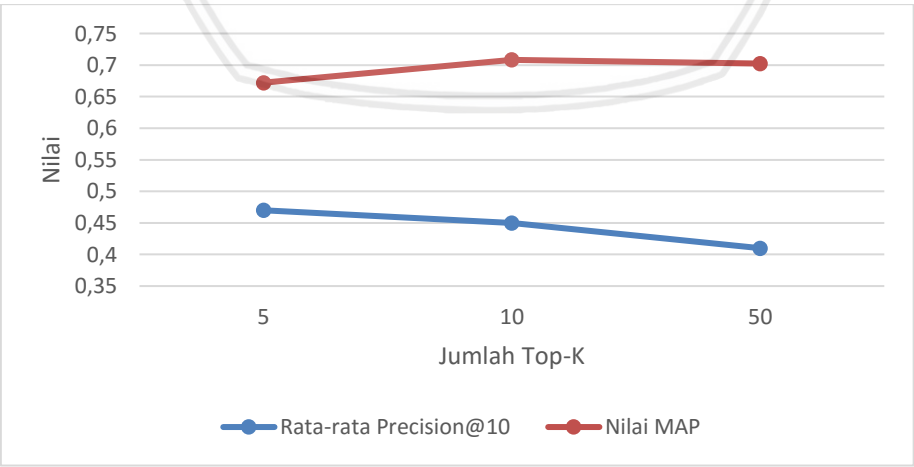
Pengujian *Top-K* kata dari hasil *Word2Vec* berdasarkan Tabel 5.5 menunjukkan hasil rata-rata nilai *precision* dari 30 *query* judul film yang masing-masing *query* menghasilkan 10 rekomendasi film. Pengujian dilakukan dengan menambahkan kata hasil *query expansion* dari *Word2Vec* sebanyak 3 kali yaitu 5 kata, 10 kata dan 50 kata, hasil menunjukkan nilai rata-rata *Precision* yang dihasilkan cenderung menurun. Hal tersebut menunjukkan bahwa semakin banyak kata yang ditambahkan melalui *query expansion,* maka kata akan semakin bervariasi dan

tidak memiliki kedekatan makna sehingga tidak cocok untuk studi kasus yang hanya menggunakan 150 data film, karena mengakhibatkan tingkat kemiripan kata akan semakin jauh dengan *query*. Nilai rata-rata *precision* terbesar yaitu 0,47, terjadi ketika pengambilan kata dari TF-IDF sebanyak 5 dan dilakukan penambahan kata untuk masing-masing kata sebanyak 5. Hasil lebih lengkap dari pengujian *Precision@K Top-K kata* dari *Word2Vec* terdapat pada Lampiran 5.

Tabel 5.6 Pengujian MAP *Top-K* Kata Hasil *Word2Vec*

Top-N kata hasil TF-IDF	Nilai MAP
5	0,7082408299
10	0,678298086
50	0,6601637797

Pengujian *Top-K* kata hasil *Word2Vec* berdasarkan Tabel 5.6 menunjukkan bahwa rata-rata nilai *Mean Average Precision* (MAP) yang dihasilkan dari 30 *query* proses rekomendasi dengan menampilkan 10 hasil rekomendasi untuk setiap *query,* menunjukkan penurunan nilai pada saat penambahan *Top-K* kata dari *Word2Vec* lebih dari I5. Hal tersebut menunjukkan bahwa rata-rata program berhasil menampilkan hasil rekomendasi dengan data yang relevan sesuai dengan peringkat kemiripan data film, namun cenderung menurun pada saat penambahan kata lebih dari 5. Semakin banyak kata yang ditambahkan, maka semakin kecil nilai MAP karena hasil kata pada *query expansion* banyak yang tidak memiliki hubungan makna, sehingga data relevan yang dihasilkan tidak sesuai peringkat semestinya. Pengujian pada tiga kali percobaan menghasilkan nilai MAP terbesar yaitu 0,7082408299 pada saat *Top-K* hasil *Word2Vec* yang ditambahkan sebanyak 5 kata. Hasil lebih lengkap dari pengujian MAP *Top-K kata* hasil *Word2Vec* terdapat pada Lampiran 6.



Gambar 5.3 Grafik Pengujian *Top-K* kata dari *Word2Vec*

5.4 Pengujian dan Analisis Tanpa *Query Expansion*

Pengujian terhadap data yang tidak melalui proses *Query Expansion* dilakukan untuk dapat membandingkan nilai yang dihasilkan. Tabel 5.5 merupakan hasil pengujian *Precision@k* dan Tabel 5.6 merupakan hasil pengujian *Mean Average Precision* (MAP) pada *Top-K* kata hasil *Word2Vec.*

Tabel 5.7 Pengujian *Precision@k* Tanpa *Query Expansion*

Jumlah *Query*	Jumlah Relevan	Jumlah Tidak Relevan	Rata-rata *Precision*@10
30	136	163	0,45333

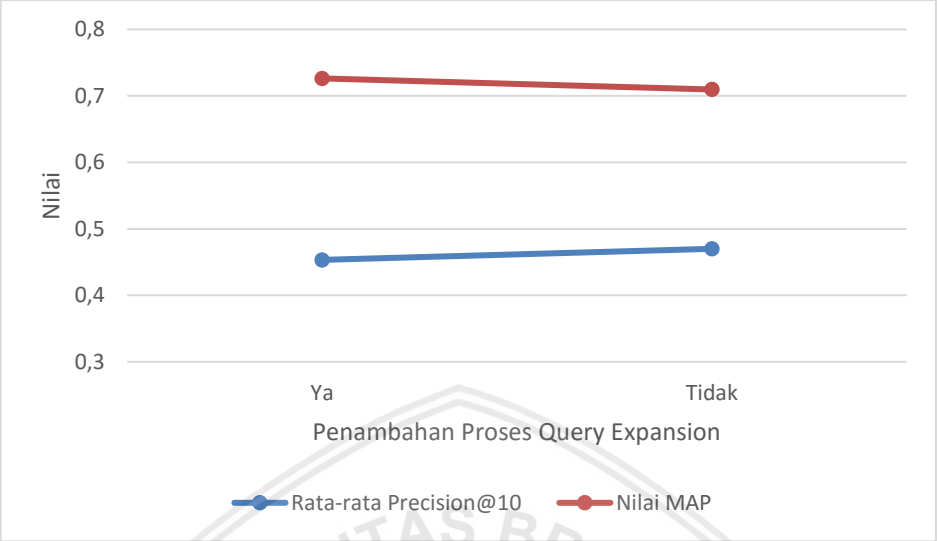
Pengujian hasil rekomendasi tanpa menggunakan proses *Query Expansion* menghasilkan nilai rata-rata *Precision*@10 sebesar 0,45333, sedangkan hasil rekomendasi menggunakan *Query Expansion* dengan metode *Word2Vec* menghasilkan nilai rata-rata *Precision@10* tertinggi yaitu sebesar 0,47 dengan parameter *hidden neuron* 10, *epoch* 110, *window size* 3, *learning rate* 0,025, *Top-N* kata hasil TF-IDF yang akan mendapat tambahan kata sebanyak 5, dan *Top-K* kata hasil *Word2Vec* yang akan ditambah ke *query* awal sebanyak 5, dapat dilihat pada Tabel 5.5. Nilai *Precision*@10 hasil rekomendasi menggunakan proses *query expansion* dengan metode *word2vec* tidak menghasilkan perbedaan nilai yang signifikan dibandingkan tidak menggunakan *query expansion.* Hal tersebut menunjukkan bahwa metode *Word2Vec* tidak menghasilkan rekomendasi film dengan data relevan yang lebih baik, karena kata yang ditambahkan banyak yang tidak memiliki hubungan kedekatan makna dengan kata pada *query* awal sehingga tidak menghasilkan variasi kata yang memiliki hubungan semantik maupun sintaksis dengan lebih baik. Hasil lebih lengkap dari pengujian *Precision@K* tanpa *Query Expansion* terdapat pada Lampiran 7.

Tabel 5.8 Pengujian MAP Tanpa *Query Expansion*

Jumlah *Query*	Nilai MAP
30	0,72626613

Pengujian hasil rekomendasi tanpa menggunakan proses *Query Expansion* berdasarkan Tabel 5.8, menunjukkan bahwa rata-rata nilai *Mean Average Precision* (MAP) dari 30 *query* proses rekomendasi dengan menampilkan 10 hasil rekomendasi untuk setiap *query* menghasilkan nilai sebesar 0,72626613, lebih besar dibandingkan nilai MAP tertinggi yang dihasilkan menggunakan proses *Query Expansion* pada perekomendasian film sebesar 0,709603, padahal nilai *Precision*@10 menggunakan *query expansion* lebih besar dibandingkan tidak menggunakan *query expansion*. Hal tersebut menunjukkan bahwa rata-rata program berhasil menampilkan hasil rekomendasi data yang relevan sesuai dengan peringkat kemiripan data film lebih banyak pada saat tanpa menggunakan *query expansion*, meskipun data relevan yang dihasilkan lebih sedikit

dibandingkan pada saat menggunakan *query expansion.* Hasil lebih lengkap dari pengujian MAP tanpa *Query Expansion* terdapat pada Lampiran 8.



Gambar 5.4 Grafik Perbandingan Hasil *Query Expansion*

Pada pengujian yang telah dilakukan sebelumnya, Tabel 5.9 menunjukkan contoh dari hasil rekomendasi yang ditampilkan untuk pencarian 1 *query*. Hasil lengkap dari proses rekomendasi terdapat pada Lampiran 11.

Tabel 5.9 Contoh Hasil Rekomendasi

Query: Harry Potter and The Goblet of Fire (2005)		
No	Hasil rekomendasi	Tag Relevansi
1	675-Harry Potter and the Order of the Phoenix (2007)	R
2	12444-Harry Potter and the Deathly Hallows: Part 1 (2010)	R
3	12445-Harry Potter and the Deathly Hallows: Part 2 (2011)	R
4	767-Harry Potter and the Half Blood Prince (2009)	R
5	338952-Fantastic Beasts: The Crimes of Grindelwald (2018)	R
6	338952-Fantastic Beasts: The Crimes of Grindelwald (2018)	R
7	363088-Ant Man and the Wasp (2018)	NR
8	287947-Shazam! (2019)	NR
9	1927-Hulk (2003)	NR
10	122-The Lord of the Rings: The Return of the King (2003)	R

Keterangan:

R = Relevan

NR = Not Relevan

BAB 6 PENUTUP

6.1 Kesimpulan

Terdapat beberapa kesimpulan yang dapat diambil berdasarkan hasil pengujian yang telah dilakukan sebelumnya, yaitu:

1. Pembuatan *Query Expansion* dengan algoritme *Word2Vec* menggunakan data latih berupa judul serta sinopsis dari film berbahasa inggris sebanyak 150 yang diambil dari halaman web MovieDb. Algoritme *Word2Vec* memproses kata perkata yang ada dalam data latih secara terurut, bobot akhir yang dihasilkan digunakan dalam membuat *Query Expansion* dengan mencari kata yang memiliki jarak tetangga terdekat untuk menghasilkan *Top-K* kata yang akan ditambahkan kedalam *query* awal. Proses rekomendasi yaitu dengan mencari nilai kedekatan antara 1 film (*query*) masukanan pengguna dengan seluruh film yang ada pada data latih dengan menghitung *cosine similarity*. Hasil pencarian sebanyak 10 film teratas akan dijadikan sebagai rekomendasi yang dihasilkan oleh sistem berdasarkan kemiripan alir sinopsis.
2. Perhitungan nilai evaluasi dilakukan dengan menggunakan *Precision@k* dan *Mean Average Precision* (MAP). Perhitungan *Precision*@10 dan *Mean Average Precision* (MAP) dilakukan untuk 30 *query* data uji dan menghasilkan rekomendasi film sebanyak 10 untuk masing-masing *query*. Hasil rata-rata nilai *Precision*@10 tertinggi yang didapatkan sebesar 0,47 menggunakan parameter pengujian *hidden neuron* sebanyak 10, *epoch* 110, *window size* 3, *learning rate* 0,025, *Top-N* kata hasil TF-IDF sebanyak 5 dan *Top-K* hasil *Word2Vec* sebanyak 5. Namun nilai tertinggi yang dihasilkan dari *Mean Average Precision* (MAP) yaitu sebesar 0,7096033, pada saat *Top-N* kata hasil TF-IDF yang diambil sebanyak 5.
3. Perhitungan nilai evaluasi hasil rekomendasi menggunakan metode *Word2Vec* untuk proses *Query Expansion* menghasilkan nilai *Precision*@10 tertinggi sebesar 0,47 dan nilai MAP tertinggi sebesar 0,7096033, sedangkan untuk proses rekomendasi tanpa menggunakan *Query Expansion* menghasilkan nilai *Precision*@10 sebesar 0,4533 dan nilai MAP sebesar 0,726266. Ketepatan sistem terhadap hasil rekomendasi tidak terdapat perbedaan yang signifikan antara menggunakan *Query Expansion* dengan tidak menggunakan *Query Expansion*, hal tersebut menunjukkan bahwa metode *Word2Vec* tidak menghasilkan rekomendasi film dengan data relevan yang lebih baik, karena penambahan kata banyak yang tidak memiliki hubungan kedekatan makna dengan kata pada *query* awal sehingga tidak menghasilkan variasi kata yang memiliki hubungan semantik maupun sintaksis dengan lebih baik. Maka, metode *Word2Vec* dalam proses *Query Expansion* untuk menghasilkan rekomendasi film dengan data latih 150, dinilai tidak cocok untuk data latih yang sedikit dan kata-kata yang kurang bervariasi karena tidak menampilkan banyak hasil yang relevan, sehingga nilai ketepatan sistem yang dihasilkan rendah.

6.2 Saran

Penelitian rekomendasi film menggunakan metode *Word2vec* sebagai algoritme pembuatan *Query Expansion* dengan 150 data latih dan 30 data uji sinopsis serta judul film belum menghasilkan hasil rekomendasi yang baik, masih terdapat banyak kekurangan yang harus diperbaiki untuk mendapatkan hasil rekomendasi film yang lebih akurat pada penelitian selanjutnya. Terdapat beberapa saran yang dapat digunakan untuk penelitian selanjutnya yaitu:

1. Melakukan penambahan data latih film dengan kata-kata yang lebih bervariasi agar dapat menghasilkan film yang memiliki kemiripan secara alir cerita dengan lebih baik.
2. Memperbanyak percobaan pengujian untuk mendapatkan nilai *hyperparameter* terbaik dalam perhitungan algoritme *Word2Vec*.
3. Menambahkan fitur lain seperti genre dan rating untuk lebih mengelompokkan dan memilih film yang memiliki tingkat relevansi dan rekomendasi yang tinggi.

DAFTAR REFERENSI

Agrawal, S. dan Jain, P., 2017. An Improved Approach for Movie Recommendation System. *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*. Palladam, India, 10-11 Februari 2017. Tersedia di < https://ieeexplore.ieee.org/document/8058367/authors> [Diakses 12 Desember 2018].

Ayu, C., Putri, A. dan Supianto, A.A., 2019. *Query Expansion Pada LINE TODAY Dengan Algoritme Extended Rocchio Relevance Feedback*. S1. Universitas Brawijaya. Tersedia di <http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/4063> [Diakses 28 April 2019]

Berry, M.W. dan Kogan, J., 2010, *Text Mining: Applications and Theory*. [e-book] United States: John Wiley & Sons. Tersedia melalui: < https://www.researchgate.net/publication/295712686_Text_Mining_Applications_and_Theory> [Diakses 12 Desember 2018].

Caselles-Dupré, H., Lesaint, F. & Royo-Letelier, J. 2018. Word2Vec applied to Recommendation: Hyperparameters Matter. *12th ACM Conference on Recommender Systems*. Vancouver, British Columbia, Canada, 2-7 Oktober 2018. New York: ACM.

"KBBI-Online", 2019. *Kamus Besar Bahasa Indonesia Online.* [online] Tersedia di < <https://kbbi.kemdikbud.go.id/entri/film> [Diakses 28 April 2019].

Laxmi, M.D., Indriati & Fauzi, M.A., 2019. *Query Expansion Pada Sistem Temu Kembali Informasi Berbahasa Indonesia Dengan Metode Pembobotan TF-IDF Dan Algoritme Cosine Similarity Berbasis Wordnet*. S1. Universitas Brawijaya. Tersedia di <http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/4223> [Diakses 28 April 2019].

Lu, Z., Shao, W. & Yu, Z., 2009, Measuring semantic similarity between words using wikipedia. *International Conference on Web Information Systems and Mining, WISM.* Shanghai, China, 7-8 November 2009. Tersedia di < https://ieeexplore.ieee.org/document/5368193> [Diakses 12 Desember 2018].

Manning, C.D., Schütze, H. dan Raghavan, P. 2008. *Introduction to information retrieval*. [e-book] New York, USA: Cambridge University Press. Tersedia di < https://nlp.stanford.edu/IR-book/pdf/irbookprint.pdf> [Diakses 23 Mei 2019].

Mikolov, T., Chen, K., Corrado, G. dan Dean, J. 2013a. Efficient Estimation of Word Representations in Vector Space. *1st International Conference on Learning Representations* (ICLR). Scottsdale, Arizona, USA, 2-4 May 2013. USA: ICLR.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. dan Dean, J. 2013b. Distributed Representations of Words and Phrases and their Compositionality. *Neural Information Processing System* (NIPS). Lake Tahor, Nevada, 5-10 Desember

2013. USA: Curran Associates Inc.

Nasution, A.W.Z., Bijaksana, M.A. dan Al Faraby, S. 2017. *Analisis dan Implementasi Perhitungan Semantics Similarity Pada Ayat Al-Quran Dengan Pendekatan Word Alignment Berdasarkan Support Vector Regression*. [*online*] Universitas Telkom. Tersedia di <http://repository.telkomuniversity.ac.id/pustaka/136787/analisis-dan-implementasi-perhitungan-semantics-similarity-pada-ayat-al-quran-dengan-pendekatan-word-alignment-berdasarkan-support-vector-regression.html> [Diakses 18 Desember 2018].

Putra, J.W.G., 2018. *Pengenalan Konsep Pembelajaran Mesin dan Deep Learning* [e-book]. Tersedia di: ResearchGate < https://www.researchgate.net/publication/323700644_Pengenalan_Pembelajaran_Mesin_dan_Deep_Learning> [Diakses 12 Desember 2018].

Rattinger, A., Le Goff, J.M. & Guetl, C., 2018. Local word embeddings for query expansion based on co-authorship and citations. in: ECIR, 2018. *Bibliometric-enhanced Information Retrieval* (BIR). Grenoble, France, 26 Maret 2018. France: CEUR.

Roy, D. & Kundu, A. 2013, *Design of Movie Recommendation System by Means of Collaborative Filtering*, [online] Tersedia di < https://pdfs.semanticscholar.org/392f/ff21ae240af751188d45f7ff56f23331442b.pdf> [Diakses 24 April 2019].

Saksono, N.D., Sari, Y.A. & Dewi, R.K., 2018. *Rekomendasi Lokasi Wisata Kuliner Menggunakan Metode K-Means Clustering Dan Simple Additive Weighting*. S1. Universitas Brawijaya. Tersedia di <http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/2766> [Diakses 24 April 2019].

Siahaan, E.J.I., Cholissodin, I. & Fauzi, M.A., 2017. *Sistem Rekomendasi Bahan Makanan Bagi Penderita Penyakit Jantung Menggunakan Algoritme Genetika*. S1.Universitas Brawijaya. Tersedia di <http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/505> [Diakses 24 April 2019].

Su, X. & Khoshgoftaar, T.M., 2009. *A Survey of Collaborative Filtering Techniques* [online] Tersedia di <https://www.researchgate.net/publication/220173171_A_Survey_of_Collaborative_Filtering_Techniques> [Diakses 12 Februari 2019].

Susilowati, E., Sabariah, M.K. & Gozali, A.A., 2015. *Implementasi Metode Support Vector Machine Untuk Melakukan Klasifikasi Kemacetan Lalu Lintas Pada Twitter Implementation Support Vector Machine Method for Traffic Jam Classification on Twitter*. S1. Universitas Telkom. Tersedia di < https://www.academia.edu/33108996/IMPLEMENTASI_METODE_SUPPORT_VECTOR_MACHINE_UNTUK_MELAKUKAN_KLASIFIKASI_KEMACETAN_LALU_LINTAS_PADA_TWITTER_IMPLEMENTATION_SUPPORT_VECTOR_MACHINE_METHOD_FOR_TRAFFIC_JAM_CLASSIFICATION_ON_TWITTER> [Diakses 24 Desember 2018].

The Movie Database. 2019. [online] Tersedia di < https://www.themoviedb.org/movie> [Diakses 26 April 2019].

Trianton, T., 2013. *Film Sebagai Media Belajar*. Yogyakarta: Graha Ilmu.

Wahyudi, I.S. 2018., *Big data analytic untuk pembuatan rekomendasi koleksi film personal menggunakan Mlib. Apache Spark.* S1. Universitas Gajah Mada. Tersedia di <https://journal.ugm.ac.id/bip/article/view/32208/21180> [Diakses 24 Desember 2019].

LAMPIRAN

Lampiran 1 Tabel Hasil Pengujian *Precision@k* untuk *Hidden Neuron*

Hidden Neuron	*Query*	Relevan	Tidak Relevan	P@10
5	Q1	5	5	0,5
	Q2	9	1	0,9
	Q3	8	2	0,8
	Q4	5	5	0,5
	Q5	5	5	0,5
	Q6	5	5	0,5
	Q7	3	7	0,3
	Q8	2	8	0,2
	Q9	1	9	0,1
	Q10	5	5	0,5
	Q11	2	8	0,2
	Q12	3	7	0,3
	Q13	3	7	0,3
	Q14	3	7	0,3
	Q15	3	7	0,3
	Q16	5	5	0,5
	Q17	6	4	0,6
	Q18	5	5	0,5
	Q19	2	8	0,2
	Q20	9	1	0,9
	Q21	4	6	0,4
	Q22	8	2	0,8
	Q23	8	2	0,8
	Q24	7	3	0,7
	Q25	7	3	0,7
	Q26	3	7	0,3
	Q27	3	7	0,3

	Q28	2	8	0,2
	Q29	4	6	0,4
	Q30	3	7	0,3
Total/ rata-rata		138	162	0,46
10	Q1	5	5	0,5
	Q2	7	3	0,7
	Q3	9	1	0,9
	Q4	4	6	0,4
	Q5	3	7	0,3
	Q6	5	5	0,5
	Q7	7	3	0,7
	Q8	3	7	0,3
	Q9	1	9	0,1
	Q10	7	3	0,7
	Q11	1	9	0,1
	Q12	7	3	0,7
	Q13	2	8	0,2
	Q14	4	6	0,4
	Q15	2	8	0,2
	Q16	5	5	0,5
	Q17	6	4	0,6
	Q18	5	5	0,5
	Q19	2	8	0,2
	Q20	7	3	0,7
	Q21	4	6	0,4
	Q22	9	1	0,9
	Q23	9	1	0,9
	Q24	8	2	0,8
	Q25	5	5	0,5
	Q26	5	5	0,5
	Q27	3	7	0,3

	Q28	3	7	0,3
	Q29	2	8	0,2
	Q30	3	7	0,3
Total/ rata-rata		141	159	0,47
20	Q1	5	5	0,5
	Q2	5	5	0,5
	Q3	7	3	0,7
	Q4	6	4	0,6
	Q5	4	6	0,4
	Q6	5	5	0,5
	Q7	3	7	0,3
	Q8	3	7	0,3
	Q9	0	10	0
	Q10	5	5	0,5
	Q11	3	7	0,3
	Q12	7	3	0,7
	Q13	2	8	0,2
	Q14	4	6	0,4
	Q15	2	8	0,2
	Q16	4	6	0,4
	Q17	4	6	0,4
	Q18	4	6	0,4
	Q19	1	9	0,1
	Q20	8	2	0,8
	Q21	4	6	0,4
	Q22	7	3	0,7
	Q23	9	1	0,9
	Q24	8	2	0,8
	Q25	4	6	0,4
	Q26	4	6	0,4
	Q27	3	7	0,3

	Q28	2	8	0,2
	Q29	3	7	0,3
	Q30	1	9	0,1
Total/ rata-rata		127	173	0,42333
50	Q1	4	6	0,4
	Q2	6	4	0,6
	Q3	7	3	0,7
	Q4	4	6	0,4
	Q5	3	7	0,3
	Q6	5	5	0,5
	Q7	0	10	0
	Q8	2	8	0,2
	Q9	1	9	0,1
	Q10	6	4	0,6
	Q11	2	8	0,2
	Q12	5	5	0,5
	Q13	2	8	0,2
	Q14	2	8	0,2
	Q15	2	8	0,2
	Q16	4	6	0,4
	Q17	3	7	0,3
	Q18	7	3	0,7
	Q19	3	7	0,3
	Q20	8	2	0,8
	Q21	7	3	0,7
	Q22	8	2	0,8
	Q23	6	4	0,6
	Q24	9	1	0,9
	Q25	5	5	0,5
	Q26	5	5	0,5
	Q27	1	9	0,1

	Q28	3	7	0,3
	Q29	2	8	0,2
	Q30	1	9	0,1
Total/ rata-rata		123	177	0,41

Lampiran 2 Tabel Hasil Pengujian MAP untuk *Hidden Neuron*

Hidden Neuron	Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
5	Q1	R	NR	NR	NR	NR	R	R	NR	R	R	
		1.0	0	0	0	0	0,333	0,428	0	0,444	0,5	0,5412
	Q2	R	R	R	R	R	NR	R	R	R	R	
		1.0	1.0	1.0	1.0	1.0	0	0,857	0,875	0,888	0,9	0,9467
	Q3	R	R	R	R	R	R	R	NR	R	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0,888	0	0,9861
	Q4	R	R	R	R	NR	NR	NR	R	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0	0,625	0	0	0,925
	Q5	R	R	NR	NR	NR	NR	R	R	NR	R	
		1.0	1.0	0	0	0	0	0,428	0,5	0	0,5	0,6857
	Q6	R	R	R	R	NR	NR	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0,714	0	0	0	0,9428
	Q7	NR	NR	NR	NR	NR	R	NR	NR	R	R	
		0	0	0	0	0	0,166	0	0	0,222	0,3	0,2296
	Q8	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
		1.0	0	0	0,5	0	0	0	0	0	0	0,75
	Q9	NR	NR	NR	NR	NR	NR	NR	NR	R	NR	
		0	0	0	0	0	0	0	0	0,111	0	0,1111
	Q10	R	R	R	R	NR	R	NR	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0,833	0	0	0	0	0,9666
	Q11	R	NR	NR	NR	NR	NR	R	NR	NR	NR	
		1.0	0	0	0	0	0	0,285	0	0	0	0,6428

Q12	NR	NR	NR	R	NR	R	R	NR	NR	NR	
	0	0	0	0,25	0	0,333	0,428	0	0	0	0,3373
Q13	R	NR	NR	NR	NR	R	NR	R	NR	NR	
	1.0	0	0	0	0	0,333	0	0,375	0	0	0,5694
Q14	NR	R	NR	R	R	NR	NR	NR	NR	NR	
	0	0,5	0	0,5	0,6	0	0	0	0	0	0,5333
Q15	NR	R	NR	NR	NR	R	R	NR	NR	NR	
	0	0,5	0	0	0	0,333	O,428	0	0	0	0,4206
Q16	R	R	R	R	NR	NR	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	0	0	0,714	0	0	0	0,9428
Q17	R	NR	NR	R	R	R	R	R	NR	NR	
	1.0	0	0	0,5	0,6	0,666	0,714	0,75	0	0	0,7051
Q18	R	NR	R	NR	R	R	NR	NR	R	NR	
	1.0	0	0,666	0	0,6	0,666	0	0	0,555	0	0,6977
Q19	NR	NR	R	NR	NR	NR	NR	NR	R	NR	
	0	0	0,333	0	0	0	0	0	0,222	0	0,2777
Q20	R	R	R	R	R	R	R	R	R	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0
Q21	R	NR	NR	R	R	R	NR	NR	NR	NR	
	1.0	0	0	0,5	0,6	0,666	0	0	0	0	0,6916
Q22	R	R	R	R	R	R	NR	R	NR	R	
	1.0	1.0	1.0	1.0	1.0	1.0	0	0,875	0	0,8	0,9593
Q23	R	R	R	R	NR	NR	R	R	R	R	
	1.0	1.0	1.0	1.0	0	0	0,714	0,75	0,777	0,8	0,8802
Q24	R	R	R	NR	R	NR	R	NR	R	R	
	1.0	1.0	1.0	0	0,8	0	0,714	0	0,666	0,7	0,8401
Q25	NR	R	R	R	NR	R	NR	R	R	R	
	0	0,5	0,666	0,75	0	0,666	0	0,625	0,666	0,7	0,6535
Q26	R	R	NR	NR	R	NR	NR	NR	NR	NR	
	1.0	1.0	0	0	0,6	0	0	0	0	0	0,8666

	Q27	NR	NR	NR	R	R	NR	NR	NR	R	NR	
		0	0	0	0,25	0,4	0	0	0	0,333	0	0,3277
	Q28	NR	R	NR	NR	NR	NR	R	NR	NR	NR	
		0	0,5	0	0	0	0	0,285	0	0	0	0,3928
	Q29	R	R	NR	R	NR	NR	NR	NR	NR	R	
		1.0	1.0	0	0,75	0	0	0	0	0	0,4	0,7875
	Q30	R	NR	NR	NR	NR	NR	R	R	NR	NR	
		1.0	0	0	0	0	0	0,285	0,375	0	0	0,5535
Nilai MAP											0,6721888017	
10	Q1	R	NR	R	NR	NR	NR	R	R	NR	NR	
		1.0	0	0,667	0	0	0	0,5	0,571	0	0	0,6476
	Q2	R	R	R	R	R	R	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	0,9571
	Q3	R	R	R	R	R	R	R	R	R	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0
	Q4	R	R	NR	R	NR	NR	NR	R	NR	NR	
		1.0	1.0	0	0,75	0	0	0	0,5	0	0	0,8125
	Q5	NR	R	NR	NR	R	NR	NR	NR	NR	R	
		0	0,5	0	0	0,4	0	0	0	0	0,3	0,3999
	Q6	R	R	R	NR	R	NR	NR	NR	R	NR	
		1.0	1.0	1.0	0	0,8	0	0	0	0,555	0	0,8111
	Q7	NR	NR	R	NR	NR	R	NR	R	R	R	
		0	0	0,333	0	0	0,33	0	0,375	0,444	0,5	0,3972
	Q8	R	NR	NR	R	R	NR	NR	NR	NR	NR	
		1.0	0	0	0,5	0,6	0	0	0	0	0	0,700
	Q9	NR	NR	NR	NR	NR	NR	NR	NR	NR	R	
		0	0	0	0	0	0	0	0	0	0,1	0,1
	Q10	R	R	NR	R	R	NR	R	R	R	NR	
		1.0	1.0	0	0,75	0,8	0	0,714	0,75	0,777	0	0,8274
	Q11	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
		1.0	0	0	0	0	0	0	0	0	0	1.0

Q12	R	R	R	R	R	R	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	1.0
Q13	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0	0	0	0	0	0	0,75
Q14	R	NR	R	NR	NR	NR	R	R	NR	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0	0	0,6488
Q15	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
	1.0	0	0	0	0	0	0	0	0,222	0	0,6111
Q16	R	R	R	R	NR	NR	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	0	0	1.0	0	0	0	0,9428
Q17	R	NR	R	R	R	NR	R	NR	NR	R	
	1.0	0	0,666	0,75	0,8	0	0,714	0	0	0,6	0,7551
Q18	R	NR	R	NR	NR	NR	R	R	R	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0,555	0	0,6301
Q19	NR	NR	R	R	NR	NR	NR	NR	NR	NR	
	0	0	0,333	0,5	0	0	0	0	0	0	0,4166
Q20	R	R	NR	NR	R	R	NR	R	R	R	
	1.0	1.0	0	0	0,6	0,66	0	0,625	0,666	0,7	0,7511
Q21	R	NR	R	NR	NR	R	NR	NR	R	NR	
	1.0	0	O,666	0	0	0,5	0	0	0,444	0	0,6527
Q22	R	R	R	R	R	NR	R	R	R	R	
	1.0	1.0	1.0	1.0	1.0	0	0,857	0,875	0,888	0,9	0,9467
Q23	R	R	R	R	R	R	R	R	R	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0
Q24	NR	R	R	R	R	R	R	R	R	NR	
	0	0,5	0,666	0,75	0,8	0,833	0,857	0,875	0,888	0	0,7713
Q25	NR	R	R	NR	R	R	NR	NR	NR	R	
	0	0,5	0,666	0	0,6	0,666	0	0	0	0,5	0,5866
Q26	R	R	R	R	NR	NR	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	0	0	0,714	0	0	0	0,9428

	Q27	NR	R	R	NR	NR	NR	NR	NR	NR	R	
		0	0,5	0,666	0	0	0	0	0	0	0,3	0,4888
	Q28	R	R	NR	NR	NR	NR	NR	NR	NR	R	
		1.0	1.0	0	0	0	0	0	0	0	0,3	0,7666
	Q29	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
		1,0	0	0	0	0	0	0	0	0,222	0	0,6111
	Q30	NR	NR	NR	NR	R	NR	NR	R	R	NR	
		0	0	0	0	0,2	0	0	0,25	0,333	0	0,2611
Nilai MAP										0,7082408299		
20	Q1	R	NR	R	NR	NR	R	NR	NR	R	R	
		1,0	0	0,666	0	0	0,5	0	0	0,444	0,5	0,6222
	Q2	R	R	R	R	NR	NR	NR	NR	R	NR	
		1,0	1,0	1,0	1,0	0	0	0	0	0,555	0	0,9111
	Q3	R	R	R	R	R	NR	NR	R	R	NR	
		1,0	1,0	1,0	1,0	1,0	0	0	0,75	0,777	0	0,9325
	Q4	R	R	R	NR	NR	R	R	R	NR	NR	
		1,0	1,0	1,0	0	0	0,666	0,714	0,75	0	0	0,8551
	Q5	R	R	NR	NR	NR	NR	R	R	NR	NR	
		1,0	1,0	0	0	0	0	0,428	0,5	0	0	0,7321
	Q6	R	R	R	R	R	NR	NR	NR	NR	NR	
		1,0	1,0	1,0	1,0	1,0	0	0	0	0	0	1.0
	Q7	NR	R	NR	NR	R	NR	R	NR	NR	NR	
		0	0,5	0	0	0,4	0	0,428	0	0	0	0,4285
	Q8	R	NR	NR	R	NR	NR	NR	NR	R	NR	
		1,0	0	0	0,5	0	0	0	0	0,333	0	0,6111
	Q9	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	
		0	0	0	0	0	0	0	0	0	0	0
	Q10	R	R	NR	NR	R	NR	NR	R	NR	R	
		1,0	1,0	0	0	0,6	0	0	0,5	0	0,5	0,72
	Q11	R	NR	NR	NR	NR	NR	NR	NR	R	R	
		1,0	0	0	0	0	0	0	0	0,222	0,3	0,5074

Q12	R	R	R	R	R	R	NR	NR	R	NR	
	1,0	1,0	1,0	1,0	1,0	1,0	0	0	1,0	0,77	0,9682
Q13	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1,0	0	0	0,5	0	0	0	0	0	0	0,75
Q14	R	R	R	NR	R	NR	NR	NR	NR	NR	
	1,0	1,0	1,0	0	0,8	0	0	0	0	0	0,95
Q15	NR	NR	NR	NR	R	NR	NR	R	NR	NR	
	0	0	0	0	0,2	0	0	0,25	0	0	0,225
Q16	R	R	R	R	NR	NR	NR	NR	NR	NR	
	1,0	1,0	1,0	1,0	0	0	0	0	0	0	1.0
Q17	R	NR	NR	R	NR	NR	R	R	NR	NR	
	1,0	0	0	0,5	0	0	0,428	0,5	0	0	0,6071
Q18	R	NR	NR	R	R	NR	NR	R	NR	NR	
	1,0	0	0	0,5	0,6	0	0	0,5	0	0	0,65
Q19	NR	NR	NR	NR	NR	NR	NR	NR	R	NR	
	0	0	0	0	0	0	0	0	0,111	0	0,1111
Q20	R	R	R	R	R	NR	NR	R	R	R	
	1,0	1,0	1,0	1,0	1,0	0	0	0,75	0,777	0,8	0,9159
Q21	R	R	NR	R	NR	NR	NR	NR	NR	R	
	1,0	1,0	0	0,75	0	0	0	0	0	0,4	0,7875
Q22	R	R	R	R	R	NR	NR	R	NR	R	
	1,0	1,0	1,0	1,0	1,0	0	0	0,75	0	0,7	0,9214
Q23	R	R	R	NR	R	R	R	R	R	R	
	1,0	1,0	1,0	0	0,8	0,833	0,857	0,875	0,888	0,9	0,9060
Q24	R	R	R	R	R	R	R	NR	R	NR	
	1,0	1,0	1,0	1,0	1,0	1,0	1,0	0	0,888	0	0,9861
Q25	R	R	NR	NR	NR	NR	NR	NR	R	R	
	1,0	1,0	0	0	0	0	0	0	0,333	0,4	0,6833
Q26	R	R	NR	R	NR	NR	NR	NR	NR	R	
	1,0	1,0	0	0,75	0	0	0	0	0	0,4	0,7875

	Q27	NR	NR	NR	NR	R	R	NR	NR	R	NR	
		0	0	0	0	0,2	0,333	0	0	0,333	0	0,2888
	Q28	NR	R	R	NR	NR	NR	NR	NR	NR	NR	
		0	0,5	0,666	0	0	0	0	0	0	0	0,5833
	Q29	R	NR	NR	NR	R	NR	R	NR	NR	NR	
		1,0	0	0	0	0,4	0	0,428	0	0	0	0,6095
	Q30	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
		1,0	0	0	0	0	0	0	0	0	0	1,0
Nilai MAP											0,7021896678	
50	Q1	R	NR	NR	NR	NR	R	NR	R	NR	R	
		1.0	0	0	0	0	0,333	0	0,375	0	0,4	0,527
	Q2	R	R	R	R	NR	R	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0,833	0,857	0	0	0	0,9484
	Q3	R	R	R	R	R	R	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	1.0
	Q4	R	R	R	R	NR	NR	NR	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0	0	0	0	1.0
	Q5	NR	NR	NR	NR	NR	R	R	R	NR	NR	
		0	0	0	0	0	0,166	0,285	0,375	0	0	0,2757
	Q6	R	R	R	R	NR	NR	NR	R	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0	0,625	0	0	0,925
	Q7	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	
		0	0	0	0	0	0	0	0	0	0	0
	Q8	R	NR	R	NR	NR	NR	NR	NR	NR	NR	
		1.0	0	0,666	0	0	0	0	0	0	0	0,8333
	Q9	NR	NR	R	NR	NR	NR	NR	NR	NR	NR	
		0	0	0,333	0	0	0	0	0	0	0	0,3333
	Q10	R	R	NR	NR	NR	R	R	R	R	NR	
		1.0	1.0	0	0	0	0,5	0,571	0,625	0,666	0	0,7271
	Q11	R	NR	NR	NR	R	NR	NR	NR	NR	NR	
		1.0	0	0	0	0,4	0	0	0	0	0	0,7

Q12	R	NR	R	NR	NR	R	NR	NR	R	R	
	1.0	0	0,666	0	0	0,5	0	0	0,444	0,5	0,6222
Q13	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0	0	0	0	0	0	0,75
Q14	NR	NR	NR	NR	NR	NR	NR	R	R	NR	
	0	0	0	0	0	0	0	0,125	0,222	0	0,1736
Q15	NR	NR	NR	NR	R	NR	NR	NR	NR	R	
	0	0	0	0	0,2	0	0	0	0	0,2	0,2
Q16	R	R	R	R	NR	NR	NR	NR	NR	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Q17	R	R	NR	NR	R	NR	NR	NR	NR	NR	
	1.0	1.0	0	0	0,6	0	0	0	0	0	0,8666
Q18	R	R	NR	R	R	R	NR	NR	R	R	
	1.0	1.0	0	0,75	0,8	0,833	0	0	0,666	0,7	0,8214
Q19	NR	R	R	NR	NR	NR	NR	NR	R	NR	
	0	0,5	0,666	0	0	0	0	0	0,333	0	0,4999
Q20	R	R	R	R	R	NR	R	NR	R	R	
	1.0	1.0	1.0	1.0	1.0	0	0,857	0	0,777	0,8	0,9293
Q21	R	NR	NR	R	R	R	R	R	R	NR	
	1.0	0	0	0,5	0,6	0,666	0,714	0,75	0,777	0	0,7155
Q22	R	R	R	R	R	R	R	NR	NR	R	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0,8	0,975
Q23	R	R	R	NR	NR	R	R	NR	R	NR	
	1.0	1.0	1.0	0	0	0,666	0,714	0	0,666	0	0,8412
Q24	R	R	R	NR	R	R	R	R	R	R	
	1.0	1.0	1.0	0	0,8	0,833	0,857	0,875	0,888	0,9	0,9060
Q25	R	NR	R	NR	NR	NR	R	R	R	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0,555	0	0,6301
Q26	R	R	NR	NR	R	NR	R	NR	R	NR	
	1.0	1.0	0	0	0,6	0	0,571	0	0,555	0	0,7453

	Q27	NR	NR	NR	NR	NR	R	NR	NR	NR	NR	
		0	0	0	0	0	0,166	0	0	0	0	0,1666
	Q28	R	NR	NR	NR	NR	R	NR	R	NR	NR	
		1.0	0	0	0	0	0,333	0	0,375	0	0	0,5694
	Q29	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
		1.0	0	0	0	0	0	0	0	0,222	0	0,6111
	Q30	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
		1.0	0	0	0	0	0	0	0	0	0	1.0
Nilai MAP											0,6764684530	

Lampiran 3 Tabel Hasil Pengujian *Precision@k Top-N* Kata Hasil TF-IDF

Top-N	*Query*	Relevan	Tidak Relevan	P@10
5	Q1	5	5	0,5
	Q2	7	3	0,7
	Q3	9	1	0,9
	Q4	4	6	0,4
	Q5	3	7	0,3
	Q6	5	5	0,5
	Q7	7	3	0,7
	Q8	3	7	0,3
	Q9	1	9	0,1
	Q10	7	3	0,7
	Q11	1	9	0,1
	Q12	7	3	0,7
	Q13	2	8	0,2
	Q14	4	6	0,4
	Q15	2	8	0,2
	Q16	5	5	0,5
	Q17	6	4	0,6
	Q18	5	5	0,5
	Q19	2	8	0,2

	Q20	7	3	0,7
	Q21	4	6	0,4
	Q22	9	1	0,9
	Q23	9	1	0,9
	Q24	8	2	0,8
	Q25	5	5	0,5
	Q26	5	5	0,5
	Q27	3	7	0,3
	Q28	3	7	0,3
	Q29	2	8	0,2
	Q30	3	7	0,3
Total/ rata-rata		141	159	0,47
10	Q1	2	8	0,2
	Q2	6	4	0,6
	Q3	10	0	1.0
	Q5	3	7	0,3
	Q6	4	6	0,4
	Q7	1	9	0,1
	Q8	3	7	0,3
	Q9	2	8	0,2
	Q10	7	3	0,7
	Q11	2	8	0,2
	Q12	7	3	0,7
	Q13	2	8	0,2
	Q14	4	6	0,4
	Q15	2	8	0,2
	Q16	5	5	0,5
	Q17	5	5	0,5
	Q18	5	5	0,5
	Q19	4	6	0,4
	Q20	6	4	0,6

	Q21	4	6	0,4
	Q22	6	4	0,6
	Q23	8	2	0,8
	Q24	9	1	0,9
	Q25	4	6	0,4
	Q26	4	6	0,4
	Q27	3	7	0,3
	Q28	2	8	0,2
	Q29	2	8	0,2
	Q30	3	7	0,3
Total/ rata-rata		129	171	0,43000

Lampiran 4 Tabel Hasil Pengujian MAP *Top-N* Kata Hasil TF-IDF

Top-N	Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
5	Q1	R	NR	R	NR	NR	NR	R	R	NR	NR	
		1.0	0	0,667	0	0	0	0,5	0,571	0	0	0,6476
	Q2	R	R	R	R	R	R	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	0,9571
	Q3	R	R	R	R	R	R	R	R	R	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0
	Q4	R	R	NR	R	NR	NR	NR	R	NR	NR	
		1.0	1.0	0	0,75	0	0	0	0,5	0	0	0,8125
	Q5	NR	R	NR	NR	R	NR	NR	NR	NR	R	
		0	0,5	0	0	0,4	0	0	0	0	0,3	0,3999
	Q6	R	R	R	NR	R	NR	NR	NR	R	NR	
		1.0	1.0	1.0	0	0,8	0	0	0	0,555	0	0,8111
	Q7	NR	NR	R	NR	NR	R	NR	R	R	R	
		0	0	0,333	0	0	0,33	0	0,375	0,444	0,5	0,3972
	Q8	R	NR	NR	R	R	NR	NR	NR	NR	NR	
		1.0	0	0	0,5	0,6	0	0	0	0	0	0,700

Q9	NR	NR	NR	NR	NR	NR	NR	NR	NR	R	
	0	0	0	0	0	0	0	0	0	0,1	0,1
Q10	R	R	NR	R	R	NR	R	R	R	NR	
	1.0	1.0	0	0,75	0,8	0	0,714	0,75	0,777	0	0,8274
Q11	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0	0	0	0	0	0	0	1.0
Q12	R	R	R	R	R	R	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	1.0
Q13	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0	0	0	0	0	0	0,75
Q14	R	NR	R	NR	NR	NR	R	R	NR	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0	0	0,6488
Q15	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
	1.0	0	0	0	0	0	0	0	0,222	0	0,6111
Q16	R	R	R	R	NR	NR	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	0	0	1.0	0	0	0	0,9428
Q17	R	NR	R	R	R	NR	R	NR	NR	R	
	1.0	0	0,666	0,75	0,8	0	0,714	0	0	0,6	0,7551
Q18	R	NR	R	NR	NR	NR	R	R	R	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0,555	0	0,6301
Q19	NR	NR	R	R	NR	NR	NR	NR	NR	NR	
	0	0	0,333	0,5	0	0	0	0	0	0	0,4166
Q20	R	R	NR	NR	R	R	NR	R	R	R	
	1.0	1.0	0	0	0,6	0,66	0	0,625	0,666	0,7	0,7511
Q21	R	NR	R	NR	NR	R	NR	NR	R	NR	
	1.0	0	O,666	0	0	0,5	0	0	0,444	0	0,6527
Q22	R	R	R	R	R	NR	R	R	R	R	
	1.0	1.0	1.0	1.0	1.0	0	0,857	0,875	0,888	0,9	0,9467
Q23	R	R	R	R	R	R	R	R	R	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0

	Q24	NR	R	R	R	R	R	R	R	R	NR	
		0	0,5	0,666	0,75	0,8	0,833	0,857	0,875	0,888	0	0,7713
	Q25	NR	R	R	NR	R	R	NR	NR	NR	R	
		0	0,5	0,666	0	0,6	0,666	0	0	0	0,5	0,5866
	Q26	R	R	R	R	NR	NR	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0,714	0	0	0	0,9428
	Q27	NR	R	R	NR	NR	NR	NR	NR	NR	R	
		0	0,5	0,666	0	0	0	0	0	0	0,3	0,4888
	Q28	R	R	NR	NR	NR	NR	NR	NR	NR	R	
		1.0	1.0	0	0	0	0	0	0	0	0,3	0,7666
	Q29	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
		1,0	0	0	0	0	0	0	0	0,222	0	0,6111
	Q30	NR	NR	NR	NR	R	NR	NR	R	R	NR	
		0	0	0	0	0,2	0	0	0,25	0,333	0	0,2611
Nilai MAP											0,7082408299	
10	Q1	R	NR	NR	NR	NR	NR	NR	R	NR	NR	
		1.0	0	0	0	0	0	0	0,25	0	0	0,625
	Q2	R	R	R	R	NR	NR	NR	R	R	NR	
		1.0	1.0	1.0	1.0	0	0	0	0,625	0,666	0	0,8819
	Q3	R	R	R	R	R	R	R	R	R	R	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	Q4	R	R	NR	R	NR	NR	NR	NR	NR	R	
		1.0	1.0	0	0,75	0	0	0	0	0	0,4	0,7875
	Q5	R	NR	NR	NR	NR	NR	NR	R	NR	R	
		1.0	0	0	0	0	0	0	0,25	0	0,3	0,5166
	Q6	R	R	R	R	NR	NR	NR	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0	0	0	0	1.0
	Q7	NR	NR	NR	R	NR	NR	NR	NR	NR	NR	
		0	0	0	0,25	0	0	0	0	0	0	0,25
	Q8	R	NR	R	NR	NR	NR	R	NR	NR	NR	
		1.0	0	0,666	0	0	0	0,428	0	0	0	0,6984

Q9	NR	NR	R	NR	NR	NR	NR	NR	R	NR	
	0	0	0,333	0	0	0	0	0	0,222	0	0,2777
Q10	R	R	R	NR	R	R	R	R	NR	NR	
	1.0	1.0	1.0	0	0,8	0,833	0,857	0,875	0	0	0,9093
Q11	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0	0	0	0	0	0	0,75
Q12	R	R	R	R	R	NR	R	R	NR	NR	
	1.0	1.0	1.0	1.0	1.0	0	0,857	0,875	0	0	0,9617
Q13	R	NR	R	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0,666	0	0	0	0	0	0	0	0,8333
Q14	NR	R	R	R	NR	NR	R	NR	NR	NR	
	0	0,5	0,666	0,75	0	0	0,571	0	0	0	0,6220
Q15	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0	0	0	0	0	0	0,75
Q16	R	R	R	R	NR	NR	NR	R	NR	NR	
	1.0	1.0	1.0	1.0	0	0	0	0,625	0	0	0,925
Q17	R	R	NR	NR	R	NR	NR	NR	R	R	
	1.0	1.0	0	0	0,6	0	0	0	0,444	0,5	0,7088
Q18	R	NR	R	R	NR	R	R	NR	NR	NR	
	1.0	0	0,666	0,7	0	0,666	0,714	0	0	0	0,7595
Q19	NR	NR	NR	NR	NR	R	R	R	R	NR	
	0	0	0	0	0	0,166	0,285	0,375	0,444	0	0,3179
Q20	R	NR	R	R	R	NR	R	NR	NR	R	
	1.0	0	0,666	0,75	0,8	0	0,714	0	0	0,6	0,7551
Q21	R	NR	NR	R	NR	R	NR	R	NR	NR	
	1.0	0	0	0,5	0	0,5	0	0,5	0	0	0,625
Q22	R	R	R	R	NR	R	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	0	0,833	0,857	0	0	0	0,9484
Q23	R	R	R	R	R	R	R	NR	R	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0,888	0	0,9861

Q24	R	R	R	NR	R	R	R	R	R	R	
	1.0	1.0	1.0	0	0,8	0,833	0,857	0,875	0,888	0,9	0,9060
Q25	R	R	NR	R	NR	NR	NR	R	NR	NR	
	1.0	1.0	0	0,75	0	0	0	0,5	0	0	0,8125
Q26	R	R	R	NR	NR	NR	NR	NR	R	NR	
	1.0	1.0	1.0	0	0	0	0	0	0,444	0	0,8611
Q27	NR	NR	NR	NR	R	NR	NR	R	R	NR	
	0	0	0	0	0,2	0	0	0,25	0,333	0	0,2611
Q28	NR	R	NR	NR	NR	NR	R	NR	NR	NR	
	0	0,5	0	0	0	0	0,285	0	0	0	0,3928
Q29	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
	1.0	0	0	0	0	0	0	0	0,222	0	0,6111
Q30	R	NR	NR	NR	NR	NR	R	R	NR	NR	
	1.0	0	0	0	0	0	0,285	0,375	0	0	0,5535
Nilai MAP										0,709603374	

Lampiran 5 Tabel Hasil Pengujian *Precision@k Top-K* Kata Hasil *Word2Vec*

Top-K	*Query*	Relevan	Tidak Relevan	P@10
5	Q1	5	5	0,5
	Q2	7	3	0,7
	Q3	9	1	0,9
	Q4	4	6	0,4
	Q5	3	7	0,3
	Q6	5	5	0,5
	Q7	7	3	0,7
	Q8	3	7	0,3
	Q9	1	9	0,1
	Q10	7	3	0,7
	Q11	1	9	0,1
	Q12	7	3	0,7
	Q13	2	8	0,2

	Q14	4	6	0,4
	Q15	2	8	0,2
	Q16	5	5	0,5
	Q17	6	4	0,6
	Q18	5	5	0,5
	Q19	2	8	0,2
	Q20	7	3	0,7
	Q21	4	6	0,4
	Q22	9	1	0,9
	Q23	9	1	0,9
	Q24	8	2	0,8
	Q25	5	5	0,5
	Q26	5	5	0,5
	Q27	3	7	0,3
	Q28	3	7	0,3
	Q29	2	8	0,2
	Q30	3	7	0,3
Total/ rata-rata		141	159	0,47
10	Q1	4	6	0,4
	Q2	4	6	0,4
	Q3	9	1	0,9
	Q4	3	7	0,3
	Q5	3	7	0,3
	Q6	5	5	0,5
	Q7	4	6	0,4
	Q8	3	7	0,3
	Q9	0	10	0
	Q10	6	4	0,6
	Q11	1	9	0,1
	Q12	7	3	0,7
	Q13	3	7	0,3

	Q14	3	7	0,3
	Q15	1	9	0,1
	Q16	6	4	0,6
	Q17	6	4	0,6
	Q18	5	5	0,5
	Q19	5	5	0,5
	Q20	5	5	0,5
	Q21	4	6	0,4
	Q22	10	0	1,0
	Q23	7	3	0,7
	Q24	8	2	0,8
	Q25	5	5	0,5
	Q26	6	4	0,6
	Q27	2	8	0,2
	Q28	3	7	0,3
	Q29	3	7	0,3
	Q30	4	6	0,4
Total/ rata-rata		135	165	0,4499
50	Q1	3	7	0,3
	Q2	7	3	0,7
	Q3	8	2	0,8
	Q4	3	7	0,3
	Q5	1	9	0,1
	Q6	5	5	0,5
	Q7	2	8	0,2
	Q8	2	8	0,2
	Q9	1	9	0,1
	Q10	4	6	0,4
	Q11	1	9	0,1
	Q12	3	7	0,3
	Q13	3	7	0,3

	Q14	3	7	0,3
	Q15	2	8	0,2
	Q16	8	2	0,8
	Q17	5	5	0,5
	Q18	5	5	0,5
	Q19	3	7	0,3
	Q20	6	4	0,6
	Q21	3	7	0,3
	Q22	10	0	1,0
	Q23	10	0	1,0
	Q24	9	1	0,9
	Q25	4	6	0,4
	Q26	3	7	0,3
	Q27	1	9	0,1
	Q28	1	9	0,1
	Q29	2	8	0,2
	Q30	5	5	0,5
Total/ rata-rata		123	177	0,4100

Lampiran 6 Tabel Hasil Pengujian MAP *Top-K* Kata Hasil *Word2Vec*

Hidden Neuron	Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
5	Q1	R	NR	R	NR	NR	NR	R	R	NR	NR	
		1.0	0	0,667	0	0	0	0,5	0,571	0	0	0,6476
	Q2	R	R	R	R	R	R	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	0,9571
	Q3	R	R	R	R	R	R	R	R	R	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0
	Q4	R	R	NR	R	NR	NR	NR	R	NR	NR	
		1.0	1.0	0	0,75	0	0	0	0,5	0	0	0,8125

Q5	NR	R	NR	NR	R	NR	NR	NR	NR	R	
	0	0,5	0	0	0,4	0	0	0	0	0,3	0,3999
Q6	R	R	R	NR	R	NR	NR	NR	R	NR	
	1.0	1.0	1.0	0	0,8	0	0	0	0,555	0	0,8111
Q7	NR	NR	R	NR	NR	R	NR	R	R	R	
	0	0	0,333	0	0	0,33	0	0,375	0,444	0,5	0,3972
Q8	R	NR	NR	R	R	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0,6	0	0	0	0	0	0,700
Q9	NR	NR	NR	NR	NR	NR	NR	NR	NR	R	
	0	0	0	0	0	0	0	0	0	0,1	0,1
Q10	R	R	NR	R	R	NR	R	R	R	NR	
	1.0	1.0	0	0,75	0,8	0	0,714	0,75	0,777	0	0,8274
Q11	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0	0	0	0	0	0	0	1.0
Q12	R	R	R	R	R	R	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	0	0	1.0
Q13	R	NR	NR	R	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0	0	0	0	0	0	0,75
Q14	R	NR	R	NR	NR	NR	R	R	NR	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0	0	0,6488
Q15	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
	1.0	0	0	0	0	0	0	0	0,222	0	0,6111
Q16	R	R	R	R	NR	NR	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	0	0	1.0	0	0	0	0,9428
Q17	R	NR	R	R	R	NR	R	NR	NR	R	
	1.0	0	0,666	0,75	0,8	0	0,714	0	0	0,6	0,7551
Q18	R	NR	R	NR	NR	NR	R	R	R	NR	
	1.0	0	0,666	0	0	0	0,428	0,5	0,555	0	0,6301
Q19	NR	NR	R	R	NR	NR	NR	NR	NR	NR	
	0	0	0,333	0,5	0	0	0	0	0	0	0,4166
Q20	R	R	NR	NR	R	R	NR	R	R	R	

		1.0	1.0	0	0	0,6	0,66	0	0,625	0,666	0,7	0,7511
	Q21	R	NR	R	NR	NR	R	NR	NR	R	NR	
		1.0	0	O,666	0	0	0,5	0	0	0,444	0	0,6527
	Q22	R	R	R	R	R	NR	R	R	R	R	
		1.0	1.0	1.0	1.0	1.0	0	0,857	0,875	0,888	0,9	0,9467
	Q23	R	R	R	R	R	R	R	R	R	NR	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0	1.0
	Q24	NR	R	R	R	R	R	R	R	R	NR	
		0	0,5	0,666	0,75	0,8	0,833	0,857	0,875	0,888	0	0,7713
	Q25	NR	R	R	NR	R	R	NR	NR	NR	R	
		0	0,5	0,666	0	0,6	0,666	0	0	0	0,5	0,5866
	Q26	R	R	R	R	NR	NR	R	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0,714	0	0	0	0,9428
	Q27	NR	R	R	NR	NR	NR	NR	NR	NR	R	
		0	0,5	0,666	0	0	0	0	0	0	0,3	0,4888
	Q28	R	R	NR	NR	NR	NR	NR	NR	NR	R	
		1.0	1.0	0	0	0	0	0	0	0	0,3	0,7666
	Q29	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
		1,0	0	0	0	0	0	0	0	0,222	0	0,6111
	Q30	NR	NR	NR	NR	R	NR	NR	R	R	NR	
		0	0	0	0	0,2	0	0	0,25	0,333	0	0,2611
Nilai MAP											0,7082408299	
10	Q1	R	NR	NR	NR	R	NR	R	R	NR	NR	
		1.0	0	0	0	0,4	0	0,428	0,5	0	0	0,5821
	Q2	R	R	R	R	NR	NR	NR	NR	NR	NR	
		1.0	1.0	1.0	1.0	0	0	0	0	0	0	1.0
	Q3	R	R	R	R	R	R	NR	R	R	R	
		1.0	1.0	1.0	1.0	1.0	1.0	0	0,875	0,888	0,9	0,9626
	Q4	R	R	NR	R	NR	NR	NR	NR	NR	NR	
		1.0	1.0	0	0,75	0	0	0	0	0	0	0,9166

Q5	NR	NR	NR	NR	NR	R	NR	NR	R	R	
	0	0	0	0	0	0,166	0	0	0,222	0,3	0,2296
Q6	NR	R	R	NR	R	NR	R	R	NR	NR	
	0	0,5	0,666	0	0,6	0	0,571	0,625	0	0	0,5926
Q7	NR	NR	R	NR	NR	R	R	NR	R	NR	
	0	0	0,333	0	0	0,333	0,428	0	0,444	0	0,3849
Q8	R	NR	NR	NR	R	NR	R	NR	NR	NR	
	1.0	0	0	0	0,4	0	0,428	0	0	0	0,6095
Q9	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	0	0	0	0	0	0	0	0	0	0	0
Q10	R	R	NR	R	R	NR	NR	NR	R	R	
	1.0	1.0	0	0,75	0,8	0	0	0	0,555	0,6	0,7842
Q11	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0	0	0	0	0	0	0	1.0
Q12	R	R	R	R	NR	R	NR	R	R	NR	
	1.0	1.0	1.0	1.0	0	0,833	0	0,75	0,777	0	0,9087
Q13	R	NR	NR	R	R	NR	NR	NR	NR	NR	
	1.0	0	0	0,5	0,6	0	0	0	0	0	0,7000
Q14	R	NR	NR	NR	R	NR	NR	NR	R	NR	
	1.0	0	0	0	0,4	0	0	0	0,333	0	0,5777
Q15	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0	0	0	0	0	0	0	1.0
Q16	R	R	R	R	R	NR	R	NR	NR	NR	
	1.0	1.0	1.0	1.0	1.0	0	0,857	0	0	0	0,9761
Q17	R	R	R	NR	NR	R	NR	R	NR	R	
	1.0	1.0	1.0	0	0	0,666	0	0,625	0	0,6	0,8152
Q18	R	NR	R	NR	R	NR	NR	R	R	NR	
	1.0	0	0,666	0	0,6	0	0	0,5	0,555	0	0,6644
Q19	NR	R	R	NR	R	NR	R	R	NR	NR	
	0	0,5	0,666	0	0,6	0	0,571	0,625	0	0	0,5926

	Q20	R	NR	NR	R	R	R	NR	NR	NR	R	
		1.0	0	0	0,5	0,6	0,666	0	0	0	0,5	0,6533
	Q21	R	R	NR	NR	NR	NR	NR	R	R	NR	
		1.0	1.0	0	0	0	0	0	0,375	0,444	0	0,7048
	Q22	R	R	R	R	R	R	R	R	R	R	
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	Q23	R	R	R	R	R	NR	R	R	NR	NR	
		1.0	1.0	1.0	1.0	1.0	0	0,857	0,875	0	0	0,9617
	Q24	NR	R	R	R	R	R	R	R	NR	R	
		0	0,5	0,666	0,75	0,8	0,833	0,857	0,875	0	0,8	0,7602
	Q25	NR	R	R	NR	R	NR	NR	R	NR	R	
		0	0,5	0,666	0	0,6	0	0	0,5	0	0,5	0,5533
	Q26	R	R	NR	R	R	NR	R	R	NR	NR	
		1.0	1.0	0	0,75	0,8	0	0,714	0,75	0	0	0,8357
	Q27	NR	NR	NR	NR	R	NR	R	NR	NR	NR	
		0	0	0	0	0,2	0	0,285	0	0	0	0,2428
	Q28	NR	NR	R	NR	R	NR	R	NR	NR	NR	
		0	0	0,333	0	0,4	0	0,428	0	0	0	0,3873
	Q29	R	NR	NR	NR	R	NR	NR	R	NR	NR	
		1.0	0	0	0	0,4	0	0	0,375	0	0	0,5916
	Q30	NR	NR	R	NR	NR	R	NR	R	NR	R	
		0	0	0,333	0	0	0,333	0	0,375	0	0,4	0,3604
Nilai MAP											0,6782980862	
50	Q1	R	NR	NR	NR	NR	NR	R	NR	R	NR	
		1.0	0	0	0	0	0	0,285	0	0,333	0	0,5396
	Q2	R	R	R	R	NR	NR	R	R	R	NR	
		1.0	1.0	1.0	1.0	0	0	0,714	0,75	0,777	0	0,8917
	Q3	R	R	R	R	NR	NR	R	R	R	R	
		1.0	1.0	1.0	1.0	0	0	0,714	0,75	0,777	0,8	0,8802
	Q4	R	NR	NR	R	R	NR	NR	NR	NR	NR	
		1.0	0	0	0,5	0,6	0	0	0	0	0	0,7000

Q5	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0	0	0	0	0	0	0	1.0
Q6	R	R	NR	R	NR	R	NR	NR	NR	R	
	1.0	1.0	0	0,75	0	0,666	0	0	0	0,5	0,7833
Q7	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
	1.0	0	0	0	0	0	0	0	0,222	0	0,6111
Q8	R	NR	NR	NR	NR	NR	NR	R	NR	NR	
	1.0	0	0	0	0	0	0	0,25	0	0	0,625
Q9	NR	NR	NR	NR	NR	NR	NR	NR	R	NR	
	0	0	0	0	0	0	0	0	0,111	0	0,111
Q10	R	R	NR	NR	R	NR	NR	NR	NR	R	
	1.0	1.0	0	0	0,6	0	0	0	0	0,4	0,75
Q11	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0	0	0	0	0	0	0	0	1.0
Q12	R	NR	NR	R	NR	NR	NR	NR	NR	R	
	1.0	0	0	0,5	0	0	0	0	0	0,3	0,6
Q13	R	NR	NR	R	NR	NR	R	NR	NR	NR	
	1.0	0	0	0,5	0	0	0,428	0	0	0	0,6428
Q14	NR	NR	R	R	NR	NR	NR	NR	NR	R	
	0	0	0,333	0,5	0	0	0	0	0	0,3	0,3777
Q15	NR	NR	NR	NR	NR	NR	R	NR	R	NR	
	0	0	0	0	0	0	0,142	0	0,222	0	0,1825
Q16	R	R	R	R	R	NR	NR	R	R	R	
	1.0	1.0	1.0	1.0	1.0	0	0	0,75	0,777	0,8	0,9159
Q17	R	NR	R	NR	R	R	NR	R	NR	NR	
	1.0	0	0,666	0	0,6	0,666	0	0,625	0	0	0,7116
Q18	R	R	R	R	NR	NR	NR	R	NR	NR	
	1.0	1.0	1.0	1.0	0	0	0	0,625	0	0	0,925
Q19	NR	R	NR	NR	R	NR	NR	NR	R	NR	
	0	0,5	0	0	0,4	0	0	0	0,333	0	0,4111

Q20	R	NR	R	R	NR	R	R	NR	NR	R	
	1.0	0	0,666	0,75	0	0,666	0,714	0	0	0,6	0,7329
Q21	R	NR	R	NR	NR	NR	NR	R	NR	NR	
	1.0	0	0,666	0	0	0	0	0,375	0	0	0,6805
Q22	R	R	R	R	R	R	R	R	R	R	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Q23	R	R	R	R	R	R	R	R	R	R	
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Q24	R	NR	R	R	R	R	R	R	R	R	
	1.0	0	0,666	0,75	0,8	0,833	0,857	0,875	0,888	0,9	0,8412
Q25	NR	R	NR	NR	NR	R	NR	R	NR	R	
	0	0,5	0	0	0	0,333	0	0,375	0	0,4	0,4020
Q26	NR	NR	R	R	NR	NR	NR	R	NR	NR	
	0	0	0,333	0,5	0	0	0	0,375	0	0	0,4027
Q27	NR	NR	NR	NR	NR	NR	R	NR	NR	NR	
	0	0	0	0	0	0	0,142	0	0	0	0,1428
Q28	NR	NR	NR	R	NR	NR	NR	NR	NR	NR	
	0	0	0	0,25	0	0	0	0	0	0	0,25
Q29	R	NR	R	NR	NR	NR	NR	NR	NR	NR	
	1.0	0	0,666	0	0	0	0	0	0	0	0,8333
Q30	R	R	R	NR	R	NR	NR	NR	NR	R	
	1.0	1.0	1.0	0	0,8	0	0	0	0	0,5	0,86
Nilai MAP											0,6601637797

Lampiran 7 Tabel Hasil Pengujian *Precision@k* Tanpa *Query Expansion*

Query	Relevan	Tidak Relevan	P@10
Q1	5	5	0,5
Q2	5	5	0,5
Q3	8	2	0,8
Q4	4	6	0,4
Q5	4	6	0,4

Q6	5	5	0,5
Q7	4	6	0,4
Q8	2	8	0,2
Q9	2	8	0,2
Q10	6	4	0,6
Q11	1	9	0,1
Q12	7	3	0,7
Q13	2	8	0,2
Q14	4	6	0,4
Q15	2	8	0,2
Q16	5	5	0,5
Q17	4	6	0,4
Q18	6	4	0,6
Q19	1	9	0,1
Q20	8	2	0,8
Q21	6	4	0,6
Q22	8	2	0,8
Q23	9	1	0,9
Q24	9	1	0,9
Q25	4	6	0,4
Q26	5	5	0,5
Q27	3	7	0,3
Q28	3	7	0,3
Q29	2	8	0,2
Q30	2	8	0,2
Total/ rata-rata	136	163	0,453

Lampiran 8 Tabel Hasil Pengujian Tanpa *Query Expansion*

Hasil rekomendasi	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10	Nilai MAP
Q1	R	NR	NR	R	R	NR	R	R	NR	NR	
	1,0	0	0	0,5	0,6	0	0,571	0,625	0	0	0,6592
Q2	R	R	R	R	R	NR	NR	NR	NR	NR	
	1,0	1,0	1,0	1,0	1,0	0	0	0	0	0	1,0
Q3	R	R	R	R	R	R	NR	R	R	NR	
	1,0	1,0	1,0	1,0	1,0	1,0	0	0,875	0,888	0	0,9704
Q4	R	R	R	NR	R	NR	NR	NR	NR	NR	
	1,0	1,0	1,0	0	0,8	0	0	0	0	0	0,95
Q5	R	NR	NR	NR	NR	NR	NR	R	R	R	
	1,0	0	0	0	0	0	0	0,25	0,333	0,4	0,4958
Q6	R	R	R	R	NR	NR	NR	R	NR	NR	
	1,0	1,0	1,0	1,0	0	0	0	0,625	0	0	0,925
Q7	NR	NR	NR	R	R	R	NR	R	NR	NR	
	0	0	0	0,25	0,4	0,5	0	0,5	0	0	0,4125
Q8	R	NR	R	NR	NR	NR	NR	NR	NR	NR	
	1,0	0	0,666	0	0	0	0	0	0	0	0,833
Q9	NR	NR	NR	NR	NR	R	R	NR	NR	NR	
	0	0	0	0	0	0,166	0,285	0	0	0	0,2261
Q10	R	R	R	R	NR	R	NR	NR	NR	R	
	1,0	1,0	1,0	1,0	0	0,833	0	0	0	0,6	0,9055
Q11	R	NR	NR	NR	NR	NR	NR	NR	NR	NR	
	1,0	0	0	0	0	0	0	0	0	0	0,1
Q12	R	R	R	R	R	R	NR	NR	R	NR	
	1,0	1,0	1,0	1,0	1,0	1,0	0	0	0,777	0	0,9682
Q13	R	NR	NR	NR	R	NR	NR	NR	NR	NR	
	1,0	0	0	0	0,4	0	0	0	0	0	0,7
Q14	NR	R	NR	NR	NR	R	NR	R	R	NR	
	0	0,5	0	0	0	0,333	0	0,375	0,444	0	0,4131

Q15	NR	R	NR	NR	R	NR	NR	NR	NR	NR	
	0	0,5	0	0	0,4	0	0	0	0	0	0,45
Q16	R	R	R	R	NR	NR	NR	NR	R	NR	
	1,0	1,0	1,0	1,0	0	0	0	0	0,555	0	0,9111
Q17	R	R	R	R	NR	NR	NR	NR	NR	NR	
	1,0	1,0	1,0	1,0	0	0	0	0	0	0	1,0
Q18	R	R	R	NR	R	NR	NR	R	R	NR	
	1,0	1,0	1,0	0	0,8	0	0	0,625	0,666	0	0,8486
Q19	NR	NR	NR	NR	NR	NR	R	NR	NR	NR	
	0	0	0	0	0	0	0,142	0	0	0	0,1428
Q20	R	R	R	R	R	NR	R	NR	R	R	
	1,0	1,0	1,0	1,0	1,0	0	0,857	0	0,777	0,8	0,9293
Q21	R	NR	NR	R	R	NR	R	NR	R	R	
	1,0	0	0	0,5	0,6	0	0,571	0	0,555	0,6	0,6378
Q22	R	R	R	R	NR	R	R	NR	R	R	
	1,0	1,0	1,0	1,0	0	0,833	0,857	0	0,777	0,8	0,9085
Q23	R	R	R	R	NR	R	R	R	R	R	
	1,0	1,0	1,0	1,0	0	0,833	0,857	0,875	0,888	0,9	0,9282
Q24	R	R	R	R	R	NR	R	R	R	R	
	1,0	1,0	1,0	1,0	1,0	0	0,857	0,875	0,888	0,9	0,9467
Q25	R	R	R	NR	NR	NR	NR	NR	R	NR	
	1,0	1,0	1,0	0	0	0	0	0	0,444	0	0,861
Q26	R	R	NR	R	NR	R	NR	NR	R	NR	
	1,0	1,0	0	0,755	0	0,666	0	0	0,555	0	0,7944
Q27	NR	NR	R	NR	R	NR	NR	R	NR	NR	
	0	0	0,333	0	0,4	0	0	0,375	0	0	0,3694
Q28	R	NR	R	NR	NR	NR	NR	NR	NR	R	
	1,0	0	0,666	0	0	0	0	0	0	0,3	0,6555
Q29	R	NR	NR	NR	NR	NR	NR	NR	R	NR	
	1,0	0	0	0	0	0	0	0	0,222	0	0,6111

Q30	NR	NR	R	NR	NR	R	NR	NR	NR	NR	
	0	0	0,333	0	0	0,333	0	0	0	0	0,3333
Total MAP										0,7262661302	

Lampiran 9 Tabel Data Latih

Dokumen	id-Judul Film (tahun)
D1	335983-Venom (2018)
D2	297802-Aquaman (2018)
D3	293660-Deadpool (2016)
D4	383498-Deadpool 2 (2018)
D5	1858-Transformers (2007)
D6	91314-Transformers: Age of Extinction (2014)
D7	335988-Transformers: The Last Knight (2017)
D8	424783-Bumblebee (2018)
D9	68726-Pacific Rim (2013)
D10	268896-Pacific Rim: Uprising (2018)
D11	297762-Wonder Woman (2017)
D12	183011-Justice League: The Flashpoint Paradox (2013)
D13	217993-Justice League: War (2014)
D14	297556-Justice League: Throne of Atlantis (2015)
D15	141052-Justice League (2017)
D16	140607-Star Wars: The Force Awakens (2015)
D17	181808-Star Wars: The Last Jedi (2017)
D18	263115-Logan (2017)
D19	287947-Shazam! (2019)
D20	155-The Dark Knight (2008)
D21	49026-The Dark Knight Rises (2012)
D22	209112-Batman v Superman: Dawn of Justice (2016)
D23	49521-Man of Steel (2013)
D24	284054-Black Panther (2018)
D25	299536-Avengers: Infinity War (2018)

D26	99861-Avengers: Age of Ultron (2015)
D27	299534-Avengers: Endgame (2019)
D28	118340-Guardians of the Galaxy (2014)
D29	283995-Guardians of the Galaxy Vol. 2 (2017)
D30	102899-Ant Man (2015)
D31	363088-Ant Man and the Wasp (2018)
D32	315635-Spider Man: Homecoming (2017)
D33	324857-SpiderMan: Into the Spider Verse (2018)
D34	272-Batman Begins (2005)
D35	1927-Hulk (2003)
D36	1724-The Incredible Hulk (2008)
D37	10195-Thor (2011)
D38	76338-Thor: The Dark World (2013)
D39	284053-Thor: Ragnarok (2017)
D40	271110-Captain America: Civil War (2016)
D41	299537-Captain Marvel (2019)
D42	1726-Iron Man (2008)
D43	10138-Iron Man 2 (2010)
D44	68721-Iron Man 3 (2013)
D45	284052-Doctor Strange (2016)
D46	351286-Jurassic World: Fallen Kingdom (2018)
D47	135397-Jurassic World (2015)
D48	338952-Fantastic Beasts: The Crimes of Grindelwald (2018)
D49	404368-Ralph Breaks the Internet (2018)
D50	121-The Lord of the Rings: The Two Towers (2002)
D51	122-The Lord of the Rings: The Return of the King (2003)
D52	338952-Fantastic Beasts: The Crimes of Grindelwald (2018)
D53	259316-Fantastic Beasts and Where to Find Them (2016)
D54	102651-Maleficent (2014)
D55	674-Harry Potter and the Goblet of Fire (2005)
D56	675-Harry Potter and the Order of the Phoenix (2007)

D57	767-Harry Potter and the Half Blood Prince (2009)
D58	12444-Harry Potter and the Deathly Hallows: Part 1 (2010)
D59	12445-Harry Potter and the Deathly Hallows: Part 2 (2011)
D60	321612-Beauty and the Beast (2017)
D61	9806-The Incredibles (2004)
D62	260513-Incredibles 2 (2018)
D63	166428-How to Train Your Dragon: The Hidden World (2019)
D64	82702-How to Train Your Dragon 2 (2014)
D65	49051-The Hobbit: An Unexpected Journey (2012)
D66	57158-The Hobbit: The Desolation of Smaug (2013)
D67	122917-The Hobbit: The Battle of the Five Armies (2014)
D68	109445-Frozen (2013)
D69	326359-Frozen Fever (2015)
D70	150540-Inside Out (2015)
D71	76492-Hotel Transylvania (2012)
D72	159824-Hotel Transylvania 2 (2015)
D73	400155-Hotel Transylvania 3: Summer Vacation (2018)
D74	177572-Big Hero 6 (2014)
D75	82690-Wreck It Ralph (2012)
D76	20352-Despicable Me (2010)
D77	324852-Despicable Me 3 (2017)
D78	93456-Despicable Me 2 (2013)
D79	585-Monsters, Inc. (2001)
D80	62211-Monsters University (2013)
D81	12536-Home Alone 4 (2003)
D82	134375-Home Alone 5: The Holiday Heist (2012)
D83	329996-Dumbo (2019)
D84	400650-Mary Poppins Returns (2018)
D85	399579-Alita: Battle Angel (2019)
D86	12155-Alice in Wonderland (2010)
D87	411-The Chronicles of Narnia: The Lion the Witch and the Wardrobe (2005)

D88	2454-The Chronicles of Narnia: Prince Caspian (2008)
D89	10140-The Chronicles of Narnia: The Voyage of the Dawn Treader (2010)
D90	424694-Bohemian Rhapsody (2018)
D91	216015-Fifty Shades of Grey (2015)
D92	337167-Fifty Shades Freed (2018)
D93	332562-A Star Is Born (2018)
D94	8966-Twilight (2008)
D95	18239-The Twilight Saga: New Moon (2009)
D96	24021-The Twilight Saga: Eclipse (2010)
D97	50619-The Twilight Saga: Breaking Dawn Part 1 (2011)
D98	50620-The Twilight Saga: Breaking Dawn Part 2 (2012)
D99	455207-Crazy Rich Asians (2018)
D100	261470-Sorry If I Call You Love (2014)
D101	576583-The Story of Us (2019)
D102	466282-To All the Boys I've Loved Before (2018)
D103	448095-I Still See You (2018)
D104	10024-My Sister's Keeper (2009)
D105	491418-Instant Family (2018)
D106	537915-After (2019)
D107	527641-Five Feet Apart (2019)
D108	37725-It's a Boy Girl Thing (2006)
D109	10625-Mean Girls (2004)
D110	200727-Love Rosie (2014)
D111	440021-Happy Death Day (2017)
D112	512196-Happy Death Day 2U (2019)
D113	439079-The Nun (2018)
D114	138843-The Conjuring (2013)
D115	259693-The Conjuring 2 (2016)
D116	77931-The Smurfs 2 (2013)
D117	396422-Annabelle: Creation (2017)
D118	346364-It (2017)

D119	49018-Insidious (2010)
D120	406563-Insidious: The Last Key (2018)
D121	91586-Insidious: Chapter 2 (2013)
D122	280092-Insidious: Chapter 3 (2015)
D123	431259-Escape Room (2017)
D124	353081-Mission: Impossible Fallout (2018)
D125	347375-Mile 22 (2018)
D126	447200-Skyscraper (2018)
D127	41513-The Smurfs (2011)
D128	584-2 Fast 2 Furious (2003)
D129	27205-Inception (2010)
D130	82992-Fast & Furious 6 (2013)
D131	168259-Furious 7 (2015)
D132	101299-The Hunger Games: Catching Fire (2013)
D133	70160-The Hunger Games (2012)
D134	291805-Now You See Me 2 (2016)
D135	75656-Now You See Me (2013)
D136	8681-Taken (2008)
D137	82675-Taken 2 (2012)
D138	260346-Taken 3 (2014)
D139	206487-Predestination (2014)
D140	157336-Interstellar (2014)
D141	302946-The Accountant (2016)
D142	262500-Insurgent (2015)
D143	44244-Camp Rock 2: The Final Jam (2010)
D144	13655-Camp Rock (2008)
D145	10947-High School Musical (2006)
D146	18126-Hannah Montana: The Movie (2009)
D147	13649-High School Musical 2 (2007)
D148	11887-High School Musical 3: Senior Year (2008)
D149	441245-High School Musical 4 (2018)

D150	313369-La La Land (2016)

Lampiran 10 Tabel Data Uji

Query	id-Judul Film (tahun)
Q1	The Chronicles of Narnia The Lion the Witch and the Wardrobe (2005)
Q2	Harry Potter and The Goblet of Fire (2005)
Q3	Iron Man (2008)
Q4	Twilight (2008)
Q5	The Incredibles (2004)
Q6	High School Musical (2006)
Q7	Despicable Me (2010)
Q8	Camp Rock 2 The Final Jam (2010)
Q9	Insidious (2010)
Q10	The Hobbit An Unexpected Journey (2012)
Q11	The Hunger Games (2012)
Q12	Now You See Me (2013)
Q13	Wreck It Ralph (2012)
Q14	Annabelle: Creation (2017)
Q15	Five Feet Apart (2019)
Q16	The Hobbit The Battle of the Five Armies (2014)
Q17	How to Train Your Dragon The Hidden World (2019)
Q18	Justice League The Flashpoint Paradox (2013)
Q19	Love Rosie (2014)
Q20	Deadpool (2016)
Q21	Aquaman (2018)
Q22	Avengers: Endgame (2019)
Q23	Avengers Infinity War (2018)
Q24	Spider Man Homecoming (2017)
Q25	Beauty and the Beast (2017)
Q26	Fantastic Beasts and Where to Find Them (2016)
Q27	Bohemian Rhapsody (2018)

Q28	The Nun (2018)
Q29	Fifty Shades of Grey (2015)
Q30	Mean Girls (2004)

Lampiran 11 Contoh Tabel Hasil Rekomendasi

Query 1: The Chronicles of Narnia The Lion the Witch and the Wardrobe (2005)		
No	Hasil Rekomendasi	Tag Relevansi
1	The Chronicles of Narnia: Prince Caspian (2008)	Relevan
2	Monsters, Inc. (2001)	Not Relevan
3	Mary Poppins Returns (2018)	Relevan
4	Frozen (2013)	Not Relevan
5	Avengers: Infinity War (2018)	Not Relevan
6	The Chronicles of Narnia: The Voyage of the Dawn Treader (2010)	Relevan
7	Maleficent (2014)	Relevan
8	Man of Steel (2013)	Not Relevan
9	Incredibles 2 (2018)	Not Relevan
10	Harry Potter and the Deathly Hallows: Part 2 (2011)	Relevan
Query 2: Harry Potter and The Goblet of Fire (2005)		
No	Hasil Rekomendasi	Tag Relevansi
1	Harry Potter and the Order of the Phoenix (2007)	Relevan
2	Harry Potter and the Deathly Hallows: Part 1 (2010)	Relevan
3	Harry Potter and the Deathly Hallows: Part 2 (2011)	Relevan
4	Harry Potter and the Half Blood Prince (2009)	Relevan
5	Fantastic Beasts: The Crimes of Grindelwald (2018)	Relevan
6	Fantastic Beasts: The Crimes of Grindelwald (2018)	Relevan
7	Ant Man and the Wasp (2018)	Not Relevan
8	Shazam! (2019)	Not Relevan
9	Hulk (2003)	Not Relevan
10	The Lord of the Rings: The Return of the King (2003)	Relevan
Query 3: Iron Man (2008)		
No	Hasil Rekomendasi	Tag Relevansi

1	Guardians of the Galaxy (2014)	Relevan
2	Iron Man 2 (2010)	Relevan
3	Batman Begins (2005)	Relevan
4	Guardians of the Galaxy Vol. 2 (2017)	Relevan
5	Iron Man 3 (2013)	Relevan
6	Spider Man: Homecoming (2017)	Relevan
7	Avengers: Age of Ultron (2015)	Relevan
8	Ant Man (2015)	Relevan
9	Star Wars: The Force Awakens (2015)	Relevan
10	How to Train Your Dragon 2 (2014)	Not Relevan
Query 4: Twilight (2008)		
No	Hasil Rekomendasi	Tag Relevansi
1	The Twilight Saga: New Moon (2009)	Relevan
2	The Twilight Saga: Breaking Dawn Part 1 (2011)	Relevan
3	Hotel Transylvania 2 (2015)	Not Relevan
4	The Twilight Saga: Eclipse (2010)	Relevan
5	Inside Out (2015)	Not Relevan
6	Instant Family (2018)	Not Relevan
7	How to Train Your Dragon: The Hidden World (2019)	Not Relevan
8	Five Feet Apart (2019)	Relevan
9	I Still See You (2018)	Not Relevan
10	Ant Man and the Wasp (2018)	Not Relevan
Query 5: The Incredibles (2004)		
No	Hasil Rekomendasi	Tag Relevansi
1	Thor: Ragnarok (2017)	Not Relevan
2	Shazam! (2019)	Relevan
3	Happy Death Day (2017)	Not Relevan
4	The Hobbit: The Desolation of Smaug (2013)	Not Relevan
5	Despicable Me (2010)	Relevan
6	Transformers (2007)	Not Relevan
7	I Still See You (2018)	Not Relevan

8	Predestination (2014)	Not Relevan
9	Camp Rock 2: The Final Jam (2010)	Not Relevan
10	Instant Family (2018)	Relevan
Query 6: High School Musical (2006)		
No	Hasil Rekomendasi	Tag Relevansi
1	High School Musical 2 (2007)	Relevan
2	High School Musical 3: Senior Year (2008)	Relevan
3	High School Musical 4 (2018)	Relevan
4	Hotel Transylvania 3: Summer Vacation (2018)	Not Relevan
5	Beauty and the Beast (2017)	Relevan
6	Insurgent (2015)	Not Relevan
7	Insidious (2010)	Not Relevan
8	Man of Steel (2013)	Not Relevan
9	La La Land (2016)	Relevan
10	Bumblebee (2018)	Not Relevan
Query 7: Despicable Me (2010)		
No	Hasil Rekomendasi	Tag Relevansi
1	High School Musical 2 (2007)	Not Relevan
2	Taken 3 (2014)	Not Relevan
3	Despicable Me 3 (2017)	Relevan
4	Iron Man 3 (2013)	Not Relevan
5	Insurgent (2015)	Not Relevan
6	Dumbo (2019)	Relevan
7	Instant Family (2018)	Not Relevan
8	Incredibles 2 (2018)	Relevan
9	Wreck It Ralph (2012)	Relevan
10	Maleficent (2014)	Relevan
Query 8: Camp Rock 2 The Final Jam (2010)		
No	Hasil Rekomendasi	Tag Relevansi
1	Camp Rock (2008)	Relevan
2	Despicable Me 3 (2017)	Not Relevan

3	Home Alone 4 (2003)	Not Relevan
4	Bohemian Rhapsody (2018)	Relevan
5	High School Musical (2006)	Relevan
6	How to Train Your Dragon 2 (2014)	Not Relevan
7	Star Wars: The Last Jedi (2017)	Not Relevan
8	Insidious: Chapter 3 (2015)	Not Relevan
9	The Incredibles (2004)	Not Relevan
10	Guardians of the Galaxy Vol. 2 (2017)	Not Relevan
Query 9: Insidious (2010)		
No	Hasil Rekomendasi	Tag Relevansi
1	After (2019)	Not Relevan
2	The Hobbit: The Battle of the Five Armies (2014)	Not Relevan
3	Harry Potter and the Half Blood Prince (2009)	Not Relevan
4	High School Musical 3: Senior Year (2008)	Not Relevan
5	A Star Is Born (2018)	Not Relevan
6	Harry Potter and the Order of the Phoenix (2007)	Not Relevan
7	Monsters, Inc. (2001)	Not Relevan
8	Harry Potter and the Deathly Hallows: Part 2 (2011)	Not Relevan
9	Hotel Transylvania 3: Summer Vacation (2018)	Not Relevan
10	The Conjuring (2013)	Relevan
Query 10: The Hobbit An Unexpected Journey (2012)		
No	Hasil Rekomendasi	Tag Relevansi
1	The Hobbit: The Desolation of Smaug (2013)	Relevan
2	The Hobbit: The Battle of the Five Armies (2014)	Relevan
3	Mile 22 (2018)	Not Relevan
4	Harry Potter and the Goblet of Fire (2005)	Relevan
5	How to Train Your Dragon 2 (2014)	Relevan
6	Inside Out (2015)	Not Relevan
7	Fantastic Beasts: The Crimes of Grindelwald (2018)	Relevan
8	Fantastic Beasts: The Crimes of Grindelwald (2018)	Relevan
9	How to Train Your Dragon: The Hidden World (2019)	Relevan

10	Fifty Shades Freed (2018)	Not Relevan
Query 11: The Hunger Games (2012)		
No	Hasil Rekomendasi	Tag Relevansi
1	The Hunger Games: Catching Fire (2013)	Relevan
2	To All the Boys I've Loved Before (2018)	Not Relevan
3	Insidious: Chapter 3 (2015)	Not Relevan
4	Fifty Shades Freed (2018)	Not Relevan
5	High School Musical 2 (2007)	Not Relevan
6	Love Rosie (2014)	Not Relevan
7	Deadpool (2016)	Not Relevan
8	Black Panther (2018)	Not Relevan
9	Iron Man 2 (2010)	Not Relevan
10	How to Train Your Dragon: The Hidden World (2019)	Not Relevan
Query 12: Now You See Me (2013)		
No	Hasil Rekomendasi	Tag Relevansi
1	Skyscraper (2018)	Relevan
2	Now You See Me 2 (2016)	Relevan
3	Taken (2008)	Relevan
4	Predestination (2014)	Relevan
5	Mile 22 (2018)	Relevan
6	Taken 3 (2014)	Relevan
7	2 Fast 2 Furious (2003)	Relevan
8	Ant Man (2015)	Not Relevan
9	Black Panther (2018)	Not Relevan
10	The Nun (2018)	Not Relevan
Query 13: Wreck It Ralph (2012)		
No	Hasil Rekomendasi	Tag Relevansi
1	Ralph Breaks the Internet (2018)	Relevan
2	Batman v Superman: Dawn of Justice (2016)	Not Relevan
3	Taken 3 (2014)	Not Relevan
4	Despicable Me 2 (2013)	Relevan

5	Man of Steel (2013)	Not Relevan
6	High School Musical 2 (2007)	Not Relevan
7	Thor (2011)	Not Relevan
8	Home Alone 4 (2003)	Not Relevan
9	The Dark Knight Rises (2012)	Not Relevan
10	Twilight (2008)	Not Relevan
Query 14: Annabelle Creation (2017)		
No	Hasil Rekomendasi	Tag Relevansi
1	The Nun (2018)	Relevan
2	Guardians of the Galaxy (2014)	Not Relevan
3	Escape Room (2017)	Relevan
4	Transformers: Age of Extinction (2014)	Not Relevan
5	The Hunger Games: Catching Fire (2013)	Not Relevan
6	Jurassic World (2015)	Not Relevan
7	Happy Death Day (2017)	Relevan
8	Happy Death Day 2U (2019)	Relevan
9	Instant Family (2018)	Not Relevan
10	Pacific Rim: Uprising (2018)	Not Relevan
Query 15: Five Feet Apart (2019)		
No	Hasil Rekomendasi	Tag Relevansi
1	To All the Boys I've Loved Before (2018)	Not Relevan
2	Despicable Me (2010)	Not Relevan
3	Hotel Transylvania 2 (2015)	Not Relevan
4	Harry Potter and the Half Blood Prince (2009)	Not Relevan
5	Justice League: The Flashpoint Paradox (2013)	Not Relevan
6	Happy Death Day 2U (2019)	Not Relevan
7	The Lord of the Rings: The Return of the King (2003)	Not Relevan
8	The Chronicles of Narnia: Prince Caspian (2008)	Not Relevan
9	Twilight (2008)	Not Relevan
10	The Nun (2018)	Relevan
Query 16: The Hobbit The Battle of the Five Armies (2014)		

No	Hasil Rekomendasi	Tag Relevansi
1	The Hobbit: The Desolation of Smaug (2013)	Relevan
2	The Hobbit: An Unexpected Journey (2012)	Relevan
3	The Lord of the Rings: The Return of the King (2003)	Relevan
4	Harry Potter and the Order of the Phoenix (2007)	Relevan
5	Justice League: War (2014)	Not Relevan
6	After (2019)	Not Relevan
7	The Twilight Saga: Breaking Dawn Part 1 (2011)	Relevan
8	Man of Steel (2013)	Not Relevan
9	Ant Man and the Wasp (2018)	Not Relevan
10	Captain Marvel (2019)	Not Relevan
Query 17: How to Train Your Dragon The Hidden World (2019)		
No	Hasil Rekomendasi	Tag Relevansi
1	How to Train Your Dragon 2 (2014)	Relevan
2	The Story of Us (2019)	Not Relevan
3	Harry Potter and the Half Blood Prince (2009)	Relevan
4	The Hobbit: An Unexpected Journey (2012)	Relevan
5	The Chronicles of Narnia: The Lion the Witch and the Wardrobe (2005)	Relevan
6	Love Rosie (2014)	Not Relevan
7	The Smurfs (2011)	Relevan
8	Maleficent (2014)	Not Relevan
9	Twilight (2008)	Not Relevan
10	The Hobbit: The Desolation of Smaug (2013)	Relevan
Query 18: Justice League The Flashpoint Paradox (2013)		
No	Hasil Rekomendasi	Tag Relevansi
1	Justice League: War (2014)	Relevan
2	After (2019)	Not Relevan
3	Alita: Battle Angel (2019)	Relevan
4	Beauty and the Beast (2017)	Not Relevan
5	Insidious: The Last Key (2018)	Not Relevan
6	Frozen Fever (2015)	Not Relevan

7	Avengers: Infinity War (2018)	Relevan
8	Justice League: Throne of Atlantis (2015)	Relevan
9	The Dark Knight Rises (2012)	Relevan
10	Alice in Wonderland (2010)	Not Relevan
Query 19: Love Rosie (2014)		
No	Hasil Rekomendasi	Tag Relevansi
1	Frozen (2013)	Not Relevan
2	Hotel Transylvania (2012)	Not Relevan
3	To All the Boys I've Loved Before (2018)	Relevan
4	The Twilight Saga: New Moon (2009)	Relevan
5	High School Musical 3: Senior Year (2008)	Not Relevan
6	Avengers: Infinity War (2018)	Not Relevan
7	Thor: Ragnarok (2017)	Not Relevan
8	The Hunger Games (2012)	Not Relevan
9	The Smurfs 2 (2013)	Not Relevan
10	Monsters, Inc. (2001)	Not Relevan
Query 20: Deadpool (2016)		
No	Hasil Rekomendasi	Tag Relevansi
1	Deadpool 2 (2018)	Relevan
2	Star Wars: The Last Jedi (2017)	Relevan
3	Beauty and the Beast (2017)	Not Relevan
4	Insidious: Chapter 3 (2015)	Not Relevan
5	Black Panther (2018)	Relevan
6	Spider Man: Homecoming (2017)	Relevan
7	My Sister's Keeper (2009)	Not Relevan
8	Avengers: Endgame (2019)	Relevan
9	Man of Steel (2013)	Relevan
10	Justice League: Throne of Atlantis (2015)	Relevan
Query 21: Aquaman (2018)		
No	Hasil Rekomendasi	Tag Relevansi
1	Justice League: Throne of Atlantis (2015)	Relevan

2	Harry Potter and the Half Blood Prince (2009)	Not Relevan
3	Justice League (2017)	Relevan
4	Interstellar (2014)	Not Relevan
5	Furious 7 (2015)	Not Relevan
6	Thor: Ragnarok (2017)	Relevan
7	Beauty and the Beast (2017)	Not Relevan
8	The Chronicles of Narnia: Prince Caspian (2008)	Not Relevan
9	Wonder Woman (2017)	Relevan
10	Maleficent (2014)	Not Relevan
Query 22: Avengers End Game (2019)		
No	Hasil Rekomendasi	Tag Relevansi
1	Captain America: Civil War (2016)	Relevan
2	Avengers: Infinity War (2018)	Relevan
3	Captain Marvel (2019)	Relevan
4	Thor: The Dark World (2013)	Relevan
5	Justice League: War (2014)	Relevan
6	The Twilight Saga: Breaking Dawn Part 2 (2012)	Not Relevan
7	Justice League (2017)	Relevan
8	Wonder Woman (2017)	Relevan
9	Spider Man: Homecoming (2017)	Relevan
10	The Dark Knight (2008)	Relevan
Query 23: Avengers Infinity War (2018)		
No	Hasil Rekomendasi	Tag Relevansi
1	Avengers: Endgame (2019)	Relevan
2	Captain America: Civil War (2016)	Relevan
3	Justice League: The Flashpoint Paradox (2013)	Relevan
4	The Dark Knight Rises (2012)	Relevan
5	Transformers (2007)	Relevan
6	Batman v Superman: Dawn of Justice (2016)	Relevan
7	Avengers: Age of Ultron (2015)	Relevan
8	Ant Man (2015)	Relevan

9	Captain Marvel (2019)	Relevan
10	Love Rosie (2014)	Not Relevan
Query 24: Spider Man Homecoming (2017)		
No	Hasil Rekomendasi	Tag Relevansi
1	Despicable Me 2 (2013)	Not Relevan
2	Avengers: Age of Ultron (2015)	Relevan
3	The Dark Knight (2008)	Relevan
4	SpiderMan: Into the Spider Verse (2018)	Relevan
5	Justice League: Throne of Atlantis (2015)	Relevan
6	Batman v Superman: Dawn of Justice (2016)	Relevan
7	Iron Man 2 (2010)	Relevan
8	The Dark Knight Rises (2012)	Relevan
9	Guardians of the Galaxy Vol. 2 (2017)	Relevan
10	Inside Out (2015)	Not Relevan
Query 25: Beauty and the Beast		
No	Hasil Rekomendasi	Tag Relevansi
1	Insidious: Chapter 3 (2015)	Not Relevan
2	High School Musical (2006)	Relevan
3	High School Musical 4 (2018)	Relevan
4	Insidious: The Last Key (2018)	Not Relevan
5	After (2019)	Relevan
6	Harry Potter and the Half Blood Prince (2009)	Relevan
7	The Lord of the Rings: The Return of the King (2003)	Not Relevan
8	Jurassic World (2015)	Not Relevan
9	The Smurfs (2011)	Not Relevan
10	High School Musical 2 (2007)	Relevan
Query 26: Fantastic Beasts and Where to Find Them (2016)		
No	Hasil Rekomendasi	Tag Relevansi
1	Fantastic Beasts: The Crimes of Grindelwald (2018)	Relevan
2	Fantastic Beasts: The Crimes of Grindelwald (2018)	Relevan
3	Harry Potter and the Deathly Hallows: Part 2 (2011)	Relevan

4	Mary Poppins Returns (2018)	Relevan
5	Pacific Rim (2013)	Not Relevan
6	The Smurfs 2 (2013)	Not Relevan
7	The Chronicles of Narnia: The Lion the Witch and the Wardrobe (2005)	Relevan
8	Escape Room (2017)	Not Relevan
9	Mile 22 (2018)	Not Relevan
10	Black Panther (2018)	Not Relevan
Query 27: Bohemian Rhapsody (2018)		
No	Hasil Rekomendasi	Tag Relevansi
1	Incredibles 2 (2018)	Not Relevan
2	Camp Rock (2008)	Relevan
3	Camp Rock 2: The Final Jam (2010)	Relevan
4	Hotel Transylvania 2 (2015)	Not Relevan
5	Big Hero 6 (2014)	Not Relevan
6	Instant Family (2018)	Not Relevan
7	The Nun (2018)	Not Relevan
8	Black Panther (2018)	Not Relevan
9	Hotel Transylvania (2012)	Not Relevan
10	High School Musical 3: Senior Year (2008)	Relevan
Query 28: The Nun (2018)		
No	Hasil Rekomendasi	Tag Relevansi
1	Annabelle: Creation (2017)	Relevan
2	Insidious: Chapter 2 (2013)	Relevan
3	After (2019)	Not Relevan
4	Monsters University (2013)	Not Relevan
5	Wonder Woman (2017)	Not Relevan
6	Man of Steel (2013)	Not Relevan
7	Monsters, Inc. (2001)	Not Relevan
8	Star Wars: The Last Jedi (2017)	Not Relevan
9	High School Musical 3: Senior Year (2008)	Not Relevan
10	The Conjuring (2013)	Relevan

Query 29: Fifty Shades of Grey (2015)		
No	Hasil Rekomendasi	Tag Relevansi
1	Fifty Shades Freed (2018)	Relevan
2	Transformers: Age of Extinction (2014)	Not Relevan
3	Happy Death Day (2017)	Not Relevan
4	The Hobbit: An Unexpected Journey (2012)	Not Relevan
5	Taken 3 (2014)	Not Relevan
6	High School Musical 3: Senior Year (2008)	Not Relevan
7	Camp Rock (2008)	Not Relevan
8	Annabelle: Creation (2017)	Not Relevan
9	La La Land (2016)	Relevan
10	Ant Man (2015)	Not Relevan
Query 30: Mean Girls (2004)		
No	Hasil Rekomendasi	Tag Relevansi
1	Insidious: Chapter 3 (2015)	Not Relevan
2	Taken (2008)	Not Relevan
3	Hotel Transylvania 3: Summer Vacation (2018)	Not Relevan
4	Home Alone 4 (2003)	Not Relevan
5	The Twilight Saga: Breaking Dawn Part 2 (2012)	Relevan
6	The Hunger Games (2012)	Not Relevan
7	Now You See Me 2 (2016)	Not Relevan
8	Instant Family (2018)	Relevan
9	Bohemian Rhapsody (2018)	Relevan
10	Annabelle: Creation (2017)	Not Relevan

Lampiran 11 menunjukkan contoh hasil rekomendasi untuk 30 *query* pada 1 kali pengujian parameter. Tabel tersebut merupakan hasil rekomendasi dengan nilai *hyperparameter hidden neuron* sebesar 10, *epoch* 110, *window size* 3, *learning rate* 0,025, *Top-N* kata hasil TF-IDF sebanyak 5 dan *Top-K* hasil *Word2Vec* sebanyak 5. Terdapat 3 kali percobaan pengujian *hidden neuron*, 2 pengujian *Top-N* kata dari TF-IDF dan 3 kali percobaan pengujian *Top-K* kata dari hasil *Word2Vec*, untuk melihat hasil pengujian menggunakan parameter lain dapat dilihat pada halaman web bit.ly/HasilRekomendasiSkripsi.