BAB II

LANDASAN TEORI

2.1 Repositori

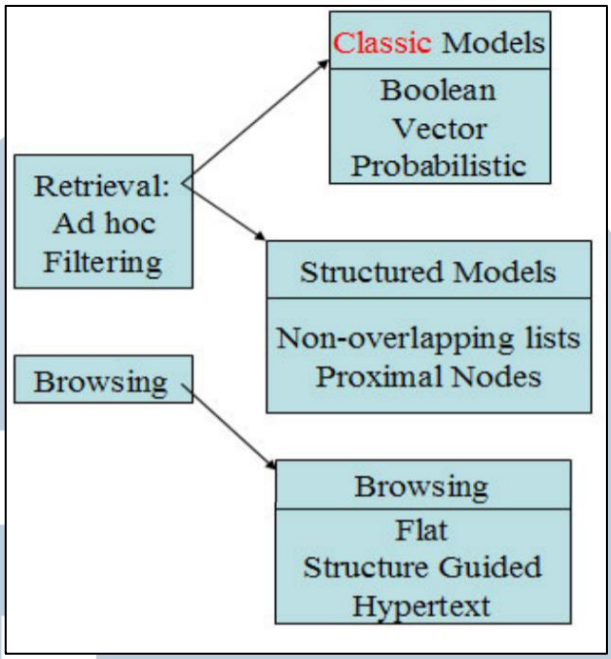
Repositori berasal dari kata *reponere* (bahasa latin) yang berarti gudang (Gaol, Irman dan Sitanggang, 2016). Secara umum repositori mengacu pada tempat terpusat dimana data disimpan dan dikelola. Sebuah repositori dapat merupakan suatu tempat dimana beberapa pangkalan data atau file ditempatkan untuk diakses melalui sebuah jaringan atau sebuah repositori data merupakan suatu lokasi yang secara langsung dapat diakses oleh pengguna (Henriyadi, 2012).

2.2 Pengembangan Piranti Lunak

Pengembangan piranti lunak merupakan tugas yang sangat kompleks. Hal tersebut membutuhkan komunikasi dengan customers, mendefinisikan tugas dan hubungan antara keduanya, membuat prediksi dan sebagainya. Dengan kata lain, yaitu sebuah rencana untuk mengembangkan sebuah piranti lunak (Zima, 2015).

2.3 Information Retrieval

Information Retrieval merupakan proses menemukan materi (biasanya dokumen) dari sebuah kumpulan data yang tidak terstruktur seperti teks untuk memenuhi kebutuhan informasi dari koleksi yang besar (Manning, Raghavan dan Schutze, 2009). Model pada *information retrieval* dapat terbagi menjadi dua kategori, yaitu *ad hoc* dan *browsing* (Lashkari, Mahdavi dan Ghomi, 2009). Adapun detil dari kategori dan relasi tersebut dapat dilihat pada Gambar 2.1.



Gambar 2.1 Kategori Information Retrieval (Lashkari, Mahdavi dan Ghomi, 2009)

2.4 Tahapan Preprocessing Data

Preprocessing merupakan kegiatan dan tahapan paling penting dalam *text mining*, *Natural Language Processing* (NLP), dan *Information Retrieval* (IR). Keputusan dalam pengembalian informasi dilakukan dengan membandingkan *terms* yang terdapat pada *query* dengan *terms* yang telah terindeks (kata atau frasa penting) yang muncul pada dokumen itu sendiri. Keputusan tersebut bisa jadi biner, yaitu antara ditolak atau diterima, atau hanya melibatkan perkiraan derajat relevansi dimana dokumen tersebut terdapat *query*. Kata-kata yang muncul pada dokumen dan *query* seringkali memiliki variasi yang terstruktur. Dikarenakan hal tersebut, sebelum melakukan *information retrieval*, tahap *preprocessing* diterapkan pada data yang digunakan untuk mengurangi ukuran data sehingga meningkatkan efektivitas pada *Information Retrieval* (Gurusamy dan Kannan, 2014). Tahapan

preprocessing data meliputi *case folding*, *tokenizing*, *filtering*, *stemming*, dan *analysing* (Handayani, 2014).

1. Case Folding

Case folding adalah proses yang digunakan untuk mengubah huruf besar menjadi huruf kecil. *Case folding* adalah proses penyamaan *case* dalam sebuah dokumen. Ini dilakukan untuk mempermudah pencarian dokumen (Leman dan Andesa, 2015).

2. Tokenizing

Tokenisasi secara garis besar dapat diartikan sebagai proses pemecahan sekumpulan karakter dalam suatu teks ke dalam satuan kata. Proses untuk membagi teks yang dapat berupa kalimat, paragraph, atau dokumen menjadi *token-token* atau bagian-bagian tertentu (Leman dan Andesa, 2015). Tokenisasi merupakan proses pemisahan suatu rangkaian karakter berdasarkan karakter spasi, dan mungkin pada waktu yang bersamaan dilakukan juga proses penghapusan karakter tertentu, seperti tanda baca. *Token* seringkali disebut sebagai istilah *term* atau kata, sebagai contoh sebuah token merupakan suatu urutan karakter dari dokumen tertentu yang dikelompokkan sebagai unit semantik yang berguna untuk diproses (Amin, 2012).

3. Filtering

Filtering adalah proses membuang kata-kata yang dianggap sebagai *noise* atau kata yang dianggap tidak penting dan tidak berpengaruh terhadap makna kata. Tahap *filtering* adalah tahap pengambilan kata-kata yang penting dari *tokenizing* (Leman dan Andesa, 2015).

4. Stemming

Proses *stemming* adalah proses untuk mencari *root* dari kata hasil dari proses *filtering*. Pencarian *root* sebuah kata atau biasa disebut dengan kata dasar dapat memperkecil hasil indeks tanpa harus menghilangkan makna. *Filtering* adalah proses pengambilan kata-kata yang dianggap penting atau mempunyai makna (Utomo, 2013).

2.5 Sastrawi

Sastrawi merupakan *library* sederhana PHP yang mengizinkan pengguna menanggalkan berimbuhan menjadi kata asalnya (*stem*). Selain kesederhanaannya, *library* Sastrawi *stemmer* merupakan *library stemmer* sederhana yang didesain untuk dapat digunakan secara mudah. Ada beberapa algoritma Sastrawi *stemmer* menerapkan algoritma yang berbasis Nazief dan Adriani, kemudian ditingkatkan oleh algoritma *Confix Stripping* (CS), kemudian ditingkatkan lagi oleh algoritma *Echanced Confix Stripping* (ECS), lalu ditingkatkan lagi oleh Modified ECS (Librian, 2014).

2.6 Pembobotan Kata (Term Weighting)

Pembobotan kata sangat berpengaruh dalam menentukan kemiripan antara dokumen dengan *query*. Apabila bobot tiap kata dapat ditentukan dengan tepat, diharapkan hasil perhitungan kemiripan teks akan menghasilkan perangkingan dokumen yang baik (Maulana, 2011). Faktor yang memegang peranan penting dalam pembobotan kata adalah sebagai berikut.

2.6.1 Term Frequency (TF)

Pendekatan dalam pembobotan lokal yang paling banyak diterapkan adalah *term frequency* (tf). Faktor ini menyatakan banyaknya kemunculan suatu kata dalam suatu dokumen. Semakin sering suatu kata muncul dalam sebuah dokumen, semakin penting kata tersebut. Ada empat cara yang bisa digunakan untuk mendapatkan nilai TF (Siregar, Sinaga dan Arianto, 2017), diantaranya yaitu:

a. Raw TF

Nilai *tf* sebuah *term* dihitung berdasarkan kemunculan *term* tersebut dalam dokumen.

b. Logarithmic TF

Dalam memperoleh nilai *tf*, cara ini menggunakan fungsi logaritmik dalam matematika. *Logarithmic tf* ditunjukkan pada Rumus 2.1.

$$TF = 1 + \log(TF) \quad \text{...(2.1)}$$

c. Binary TF

Cara ini menghasilkan nilai *boolean* berdasarkan kemunculan *term* pada dokumen tersebut. Akan bernilai 0 apabila *term* tidak ada pada sebuah dokumen, dan bernilai 1 apabila *term* tersebut ada pada dokumen. Sehingga banyaknya kemunculan *term* pada dokumen tidak berpengaruh.

d. Augmented TF

Nilai *tf* adalah jumlah kemunculan *term* pada sebuah dokumen. Nilai max(TF) adalah jumlah kemunculan terbanyak *term* pada dokumen yang sama. Adapun *augmented tf* ditunjukkan pada Rumus 2.2.

$$TF = 0.5 + 0.5\ x\ \text{TFmax}(TF) \quad \text{...(2.2)}$$

2.6.2 Inverse Document Frequency (IDF)

Pembobotan global digunakan untuk memberikan tekanan terhadap *term* yang mengakibatkan perbedaan dan berdasarkan pada penyebaran dari *term* tertentu di seluruh dokumen. Banyak skema didasarkan pada pertimbangan bahwa semakin jarang suatu *term* muncul di dalam total koleksi maka menjadi semakin berbeda (Rozanda, Marsal dan Iswanti, 2013). Bobot global dari suatu *term* yang ke-*i* pada pendekatan *inverse document frequency* (idf_i) dapat dirumuskan seperti pada Rumus 2.3.

$$idf_{(i)} = \log(\frac{N}{Df_{(i)}}) \quad ...(2.3)$$

Keterangan:

a. N adalah jumlah artikel dalam koleksi dokumen

b. $Df_{(i)}$ adalah jumlah dokumen koleksi yang mengandung *term* yang ke-*i*

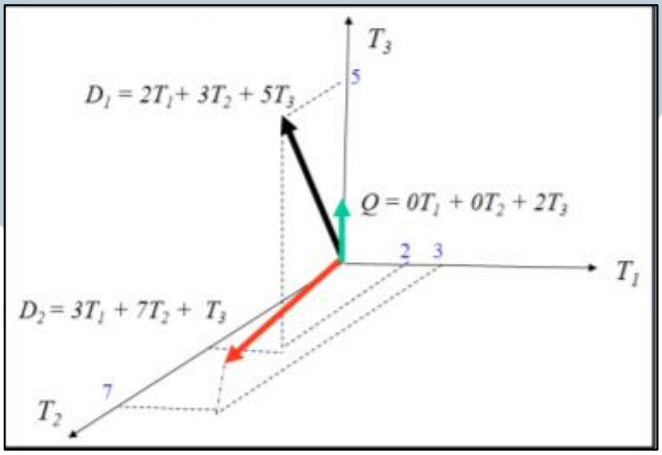
Bobot *term* yang ke-*i* di dalam *Information Retrieval System* (w_{ij}) dihitung menggunakan *tf-idf* yang ditunjukkan pada Rumus 2.4

$$w_{ij} = tf\ x\ idf \quad ...(2.4)$$

2.7 Vector Space Model

Vector Space Model atau *Term Vector Model* adalah sebuah model aljabar untuk menggambarkan dokumen teks (beberapa objek) sebagai vektor dari *identifier*. Biasanya digunakan dalam penyaringan informasi (*information filtering*), penemuan informasi (*information retrieval*), *indexing*, dan *ranking* yang saling relevan (Indranandita, Susanto dan Rahmat, 2008). Model ini akan menghitung derajat kesamaan antara setiap dokumen yang disimpan di dalam

sistem dengan *query* yang diberikan oleh pengguna (Nadirman, 2006). Pada *Vector Space Model*, setiap dokumen dan kata kunci dari pengguna direpresentasikan sebagai ruang vektor berdimensi n. kata kunci dan dokumen dianggap sebagai vektor-vektor pada ruang n-dimensi. Penentuan relevansi dokumen dengan kata kunci dipandang sebagai pengukuran kesamaan (*similarity measure*) antara vektor dokumen dengan vektor kata kunci. Semakin sama suatu vektor dokumen dengan vektor kata kunci maka dokumen dapat dipandang semakin relevan dengan kata kunci. Gambar 2.1 menunjukkan contoh dari *Vector Space Model*.



Gambar 2.1 Vector Space Model (Annisa, Nengsih dan Ananda, 2014)

Keterangan:

a. D adalah dokumen
b. Q adalah query
c. T adalah *term* yang menjadi dimensi dari VSM

Beberapa tahap dalam mengimplementasikan *Vector Space Model* yaitu melakukan pembobotan kata dengan TF-IDF, menghitung jarak *query*, menghitung jarak dokumen, melakukan perhitungan *inner product* antara *query* dan dokumen,

menghitung similaritas dengan *cosine similarity*, dan membuat *ranking*. Adapun rumus perhitungan jarak pada *query* ditunjukkan pada Rumus 2.5 (Amin, 2013).

$$| q | = \sqrt{\sum_{j=1}^{t}(W_{iq})^2} \quad ...(2.5)$$

Keterangan:

a. | q | adalah jarak *query*

b. W_{iq} adalah bobot *query* dokumen ke-*i*

Jarak *query* dihitunng untuk didapatkan jarak *query* dari bobot *query* dokumen yang terambil oleh sistem. Adapun jarak dokumen didapatkan berdasarkan Rumus 2.6 (Amin, 2013).

$$| d_j | = \sqrt{\sum_{i=1}^{t}(W_{ij})^2} \quad ...(2.6)$$

Keterangan:

c. $| d_j |$ adalah jarak dokumen

d. W_{ij} adalah bobot dokumen ke-*i*

Jarak dokumen dihitung untuk didapatkan jarak dokumen dari bobot dokumen-dokumen yang terambil oleh sistem.

2.8 Cosine Similarity

Cosine Similarity digunakan untuk mengevaluasi tingkat similaritas atau kemiripan dari dokimen (dj), berkaitan dengan query (q) sebagai korelasi antara vektor dj dan q (Amin dan Purwaningtyas, 2015). Menggunakan *cosine similarity* merupakan cara yang baik untuk melakukan *rank* pada dokumen dengan mencari

dokumen mana yang lebih dekat dengan *query* yang dimasukkan *user* (Salih, 2018). Perhitungan *cosine similarity* ditunjukkan oleh Rumus 2.7.

$$similarity\ (\vec{d}_j.\vec{q}) = \frac{\vec{d}_j\ .\ \vec{q}}{|\vec{d}_j|\ .\ |\vec{q}|} = \frac{\sum_{i=1}^{t}(w_{ij}\ .\ w_{iq})}{\sqrt{\sum_{i=1}^{t}(w_{ij})^2}\ .\sqrt{\sum_{i=1}^{t}(w_{iq})^2}} \quad ...(2.7)$$

Keterangan:

a. $similarity\ (\vec{d}_j.\vec{q})$ adalah similaritas antara *query* dan dokumen

b. $|\vec{d}_j|$ adalah jarak dokumen

c. $|\vec{q}|$ adalah jarak *query*

d. w_{ij} adalah bobot dokumen ke-*i*

e. w_{iq} adalah bobot *query* dokumen ke-*i*

2.9 Precision, Recall, dan F-measure

Untuk mengukur kualitas dari dokumen yang terambil, maka diperlukan perhitungan relevansi dengan *precision* dan *recall*. *Precision* mengevaluasi kemampuan sistem *Information Retrieval* untuk menemukan kembali *top-ranked* yang paling relevan, sedangkan *recall* mengevaluasi kemampuan sistem untuk menemukan semua *item* yang relevan dari dalam koleksi dokumen (Rozanda, Marsal dan Iswanti, 2013). Perolehan (*recall*) berhubungan dengan kemampuan sistem untuk memanggil dokumen yang relevan dengan *query*, sedangkan ketepatan (*precision*) berkaitan dengan kemampuan sistem untuk tidak memanggil dokumen yang tidak relevan dengan kebutuhan pengguna. Dokumen terpanggil (*recall*) yang relevan dengan *query* pengguna belum tentu relevan dengan kebutuhan pengguna. Rasio dari tingkat perolehan (*recall*) dan ketepatan (*precision*) yang dicapai dalam kegiatan penelusuran dapat diungkapkan sebagai

berikut (Hasugian, 2006). *recall* ditunjukkan pada Rumus 2.8, sedangkan *precision* ditunjukkan pada Rumus 2.9.

$$recall = \frac{Jumlah\ dokumen\ relevan\ yang\ terpanggil}{Jumlah\ dokumen\ relevan\ yang\ ada\ dalam\ database} \quad ...(2.8)$$

$$precision = \frac{Jumlah\ dokumen\ yang\ terpanggil\ relevan\ dengan\ kebutuhan}{Jumlah\ dokumen\ yang\ terpanggil\ oleh\ sistem} \quad ...(2.9)$$

Contingency Table untuk melakukan perhitungan *precision* dan *recall* ditunjukkan pada Tabel 2.1.

Tabel 2.1 Contingency Table (Hasugian, 2006)

	Relevan	Tidak Relevan	Total
Ditemukan	A	B	A + B
Tidak Ditemukan	C	D	C + D
Total	A + C	B + D	A+B+C+D

Berdasarkan Tabel 2.1, perhitungan dapat dilakukan dengan mengacu pada rasio yang telah dikemukakan sebelumnya. Untuk menghitung rasio *recall*, terlebih dahulu kita tentukan jumlah dokumen relevan yang terambil, berdasarkan data pada tabel yaitu A, sedangkan jumlah dokumen relevan yang ada dalam *database* adalah A+C. Selanjutnya untuk menghitung *precision*, hal yang sama pada perhitungan *recall* diberlakukan yaitu jumlah dokumen relevan yang terambil adalah A, sedangkan jumlah dokumen yang terambil dalam pencarian adalah A+B.

F-measure merupakan standar pengukuran untuk melakukan evaluasi terhadap hasil *information retrieval* (Kandefer dan Saphiro, 2009). *F-measure* adalah *harmonic mean* atau nilai rataan harmonis antara perhitungan *precision* dan *recall*.

Perhitungan *f-measure* ditunjukkan pada Rumus 2.10 (Manning, Raghavan dan Schutze, 2009).

$$f - measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad ...(2.10)$$

2.10 Average Precision dan Mean Average Precision

Average Precision merupakan pengukuran yang dirancang untuk mengevaluasi algoritma *Information Retrieval* (Zhu, 2004). Untuk setiap kebutuhan informasi, *Average Precision* adalah rata-rata dari nilai *precision* yang diperoleh pada sekumpulan *k* dokumen teratas yang muncul setelah semua dokumen relevan terambil (Manning, Raghavan dan Schutze, 2009). Adapun *Mean Average Precision* (MAP) merupakan nilai rata-rata dari *average precision*. Rumus 2.11 menunjukkan rumus untuk menghitung nilai *Mean Average Precision*.

$$MAP\ (Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \quad ...(2.11)$$

Keterangan:

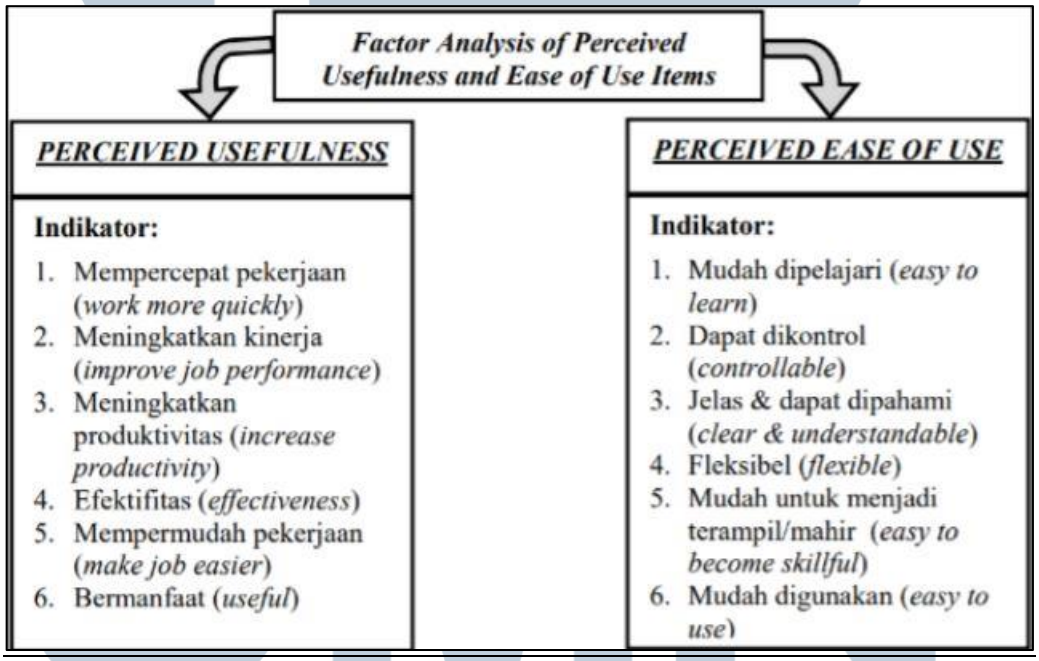
a. Q adalah jumlah *query* uji
b. R adalah *item* relevan yang dihasilkan oleh sistem
c. m adalah jumlah *item* relevan yang dihasilkan dari *query*

2.11 Technology Acceptance Model

Technology Acceptance Model (TAM) merupakan model yang mengkonsepkan bagaimana pengguna menerima dan menggunakan teknologi baru. Asal dari pendekatan teori psikologis untuk menjelaskan pengguna yang mengacu pada kepercayaan, sikap, minat, dan hubungan perilaku pengguna. Ciri khas dari

model TAM adalah sederhana namun bisa memprediksi penerimaan maupun penggunaan teknologi (Fatmawati, 2015).

Berdasarkan TAM, ada dua faktor yang mempengaruhi asumsi pengguna saat menggunakan sistem informasi yang baru, yaitu *perceived ease of use* dan *perceived usefulness*. *Perceived ease of use* merujuk pada sejauh mana seseorang percaya bahwa menggunakan suatu sistem tertentu akan bebas dari usaha, sedangkan *perceived usefulness* merujuk kepada sejauh mana seseorang percaya bahwa menggunakan sistem tertentu akan meningkatkan pekerjaannya (Davis, 1989).



Gambar 2.2 Factor Analysis of Perceived Usefulness and Ease of Use Questions (Davis, 1989)

2.12 Skala Likert

Skala Likert adalah suatu skala psikometrik yang umum digunakan dalam kuesioner, dan merupakan skala yang palin banyak digunakan dalam riset berupa

survei. Sewaktu menanggapi pertanyaan dalam skala Likert, responden menentukan persetujuan mereka terhadap suatu pernyataan dengan memilih salah satu dari pilihan yang tersedia (Syofian, Setiyaningsih dan Syamsiah, 2015). Biasanya disediakan lima pilihan skala dengan format seperti:

1. Sangat setuju
2. Setuju
3. Netral
4. Tidak Setuju
5. Sangat Tidak Setuju