ISSN 2085-4552

Aplikasi Rekomendasi Buku pada Katalog Perpustakaan Universitas Multimedia Nusantara Menggunakan Vector Space Model

Richard Firdaus Oeyliawan[1], Dennis Gunawan[2]
Jurusan Teknik Informatika Fakultas Teknik dan Informatika Universitas Multimedia Nusantara
richfir@gmail.com
dennis.gunawan@umn.ac.id

Diterima 16 Oktober 2017
Disetujui 20 Desember 2017

***Abstract*—Library is one of the facilities which provides information, knowledge resource, and acts as an academic helper for readers to get the information. The huge number of books which library has, usually make readers find the books with difficulty. Universitas Multimedia Nusantara uses the Senayan Library Management System (SLiMS) as the library catalogue. SLiMS has many features which help readers, but there is still no recommendation feature to help the readers finding the books which are relevant to the specific book that readers choose. The application has been developed using Vector Space Model to represent the document in vector model. The recommendation in this application is based on the similarity of the books description. Based on the testing phase using one-language sample of the relevant books, the F-Measure value gained is 55% using 0.1 as cosine similarity threshold. The books description and variety of languages affect the F-Measure value gained.**

***Index Terms*—Book Recommendation, Porter Stemmer, SLiMS Universitas Multimedia Nusantara, TF-IDF, Vector Space Model**

I. PENDAHULUAN

Perpustakaan merupakan salah satu fasilitas penyedia informasi, sumber ilmu pengetahuan, dan sarana penunjang proses kegiatan belajar mengajar bagi para pengguna untuk mendapatkan informasi yang diinginkan [1]. Jumlah buku yang banyak pada sebuah perpustakaan universitas membuat pembaca kerap kali kesulitan dalam menemukan buku yang diinginkan. Selain itu, mencari buku yang memiliki kesamaan dengan buku yang dipilih sebelumnya, akan sulit dilakukan di tengah banyaknya buku-buku di perpustakaan tersebut [2].

Perpustakaan Universitas Multimedia Nusantara (UMN) memiliki visi untuk menjadi learning resource center dalam memenuhi kebutuhan informasi dan pengetahuan *civitas academica* yang berbasis ICT [3]. Sistem pencarian buku pada katalog perpustakaan UMN saat ini masih menggunakan Online Public Access Catalog (OPAC) dengan sistem Senayan Library Management System (SLiMS) yang diakses melalui website. SLiMS adalah sebuah sistem yang dapat digunakan secara open source untuk pencarian koleksi buku berbasis PHP dan MySQL [4].

SLiMS memiliki beberapa modul yang membantu pengguna seperti modul penelusuran, modul sirkulasi, modul manajemen keanggotaan, modul inventaris koleksi, dan modul pengatalogan. Namun, fitur rekomendasi koleksi masih belum dihadirkan dalam sistem SLiMS [4].

Peminjam buku pada perpustakaan membutuhkan alternatif informasi buku lain ketika buku yang hendak dipinjam sedang dipinjam [5]. Referensi peminjam buku dapat menjadi lebih luas jika sistem pencarian ditambahkan fitur rekomendasi buku.

Sistem rekomendasi merupakan *subclass* dari sistem information filtering yang memprediksi rating atau preferensi oleh pengguna terhadap suatu item [6]. Terdapat beberapa metode yang dijadikan dasar dari sistem rekomendasi, termasuk metode yang populer, seperti Collaborative Filtering, dan Content-based [6]. Content-based filtering memberikan rekomendasi berdasarkan deskripsi dari item [7]. Berbeda dengan pendekatan Content-based Filtering, Collaborative Filtering merekomendasikan item dengan mencari kemiripan selera seorang pengguna dengan pengguna lain. Pendekatan Content-based Filtering bekerja dengan melihat kemiripan item baru dengan item yang telah dipilih sebelumnya sehingga sistem tidak memerlukan rating dari pengguna lain [8]. Kemiripan item dapat dianalisis dari feature yang dikandung oleh item sebelumnya seperti deskripsi item, audio segments [9], genre dan sinopsis film [10].

Terdapat beberapa metode yang termasuk dalam pendekatan Content-based, salah satunya adalah teknik pembobotan term yang terdapat pada deskripsi item dan melakukan perhitungan *similarity* dengan deskripsi item lain [11]. Teknik pembobotan yang umum digunakan adalah TF-IDF dengan Vector Space Model sebagai metode untuk melakukan pengukuran kemiripan dengan cosine similarity.

Sebelumnya, terdapat penelitian mengenai implementasi metode *Vector Space Model* pada sistem rekomendasi adalah penelitian yang dilakukan oleh [12]. Sistem rekomendasi yang dibangun berfungsi untuk memberikan rekomendasi karya tulis yang melibatkan judul dari paper selama proses perhitungan kemiripan, baik kemiripan antara query dengan daftar paper maupun kemiripan antara satu paper dengan paper lainnya. Dalam penelitian tersebut, perhitungan kemiripan antar *query Q* dengan beberapa dokumen *Di* menggunakan *cosine similarity*. Teknik pembobotan term yang digunakan adalah term frequency (tf) dan inverse document frequency (idf). Dari hasil penelitian yang telah dilakukan, dikatakan bahwa pemanfaatan sistem rekomendasi yang dibangun sangat membantu pengguna dalam mendapatkan karya tulis yang sesuai dengan kebutuhan.

Oleh karena itu, berdasarkan permasalahan yang ada dan penelitian terkait mengenai masalah sejenis, akan diimplementasikan metode *Content-based Filtering* menggunakan *Vector Space Model*, algoritma TF-IDF, dan *cosine similarity* untuk menghitung kesamaan deskripsi antar buku.

II. LANDASAN TEORI

A. *Text Mining*

Text mining digunakan untuk mendeskripsikan teknik dari data mining yang secara otomatis menemukan sesuatu hal yang berguna atau sebuah pengetahuan baru dari sebuah teks yang tidak terstruktur [13]. Kunci dari proses ini adalah menggabungkan informasi yang berhasil diekstraksi dari berbagai sumber [14]. Tujuan dari *text mining* yaitu mendapatkan informasi yang bermanfaat dari kumpulan dokumen yang ada. Selain itu, *text mining* dapat membantu permasalahan seperti pemrosesan, pengorganisasian atau pengelompokan dan menganalisis teks yang tidak terstruktur dalam jumlah besar [15].

B. *Tahapan Preprocessing Data*

Teks yang akan dilakukan proses *text mining*, pada umumnya memiliki beberapa karakteristik yaitu memiliki dimensi yang tinggi, terdapat *noise* pada data, dan terdapat struktur teks yang tidak baik. Oleh karena itu, pada proses *text mining*, terdapat beberapa tahapan awal (*preprocessing*) yang perlu dilakukan, yaitu *case folding*, *tokenizing*, *filtering*, *stemming*, dan *analyzing* [15].

1. Case Folding

Case folding adalah proses yang pertama kali dilakukan dalam rangkaian perancangan klasifikasi dokumen teks. Proses ini merupakan proses di mana kata-kata di dalam dokumen atau kalimat akan diubah menjadi huruf kecil (a sampai z) dan menghilangkan tanda baca. Karakter lain selain huruf akan dianggap *delimiter* sehingga karakter tersebut akan dihilangkan. Hal ini dilakukan untuk mencegah terjadinya *noise* pada saat pengambilan informasi. Untuk selanjutnya, hasil dari *case folding* nantinya akan digunakan untuk proses *tokenizing*.

2. Tokenizing

Proses *tokenizing* merupakan proses yang dilakukan setelah melakukan proses *case folding*. Pada tahap ini dilakukan pemotongan *string input* berdasarkan tiap kata yang menyusunnya. Hasil pemrosesan akan berupa kata yang disebut dengan *token*/*term*. *Term* ini nantinya akan disimpan ke dalam *database* untuk dilakukan *indexing* saat melakukan pencarian.

3. Filtering

Filtering atau *parsing* merupakan proses pengambilan kata-kata penting dari hasil *token*. Tahap *filtering* dapat dilakukan dengan menghapus *stoplist/stopword* (membuang kata yang kurang penting). *Stopword* adalah kata-kata yang sering muncul dalam teks dalam jumlah besar dan dianggap tidak memiliki makna. Pada tahap ini, kata-kata yang merupakan *stopword* akan dihilangkan. *Stopword* ini dapat berupa kata penghubung, kata depan, dan kata pengganti, contohnya seperti “yang”, “di”, “dan”, “ke”, dan “dari”. Tujuan dari proses ini adalah mengurangi volume kata sehingga hanya kata-kata penting saja yang terdapat pada dokumen

4. Stemming

Proses stemming merupakan proses untuk mencari kata dasar dari kata yang sudah mengalami proses *filtering*. Pencarian kata dasar sebuah kata dapat memperkecil hasil indeks tanpa harus menghilangkan makna. Proses *stemming* dilakukan dengan menghilangkan semua imbuhan baik yang terdiri dari awalan (*prefix*), akhiran (*suffix*), sisipan (*infix*), bentuk perulangan, dan kombinasi antara awalan dan akhiran (*confix*). Tujuan dari proses ini adalah mengurangi variasi kata yang mempunyai kata dasar yang sama.

C. *Term Frequency-Inverse Document Frequency (TF-IDF)*

Term Frequency-Inverse Document Frequency merupakan teknik dalam memberikan bobot hubungan suatu term terhadap sebuah dokumen [16]. TF-IDF bekerja dengan memberikan bobot pada suatu *term* dalam sebuah dokumen [17]. Metode ini bekerja dengan menggabungkan dua konsep untuk perhitungan bobot, yaitu frekuensi kemunculan sebuah *term* di dalam sebuah dokumen (*tf*) dan inversi frekuensi dokumen (*idf*) yang mengandung kata tersebut [18]. Frekuensi kemunculan kata di dalam dokumen yang diberikan menunjukkan seberapa penting kata tersebut di dalam dokumen tersebut [17]. Jumlah frekuensi dokumen yang mengandung kata tersebut menunjukan seberapa umum kata tersebut. Semakin sedikit jumlah dokumen yang mengandung *term* yang dimaksud maka nilai pada *idf* akan semakin besar [19]. Berikut persamaan untuk penghitungan *term frequency* [20].

$$tf_{i,j} = f_{i,j} \quad (1)$$

Dengan *tf* adalah *term frequency*, dan $tf_{i,j}$ adalah banyaknya kemunculan term t_i dalam dokumen dj, *Term frequency* (*tf*) dihitung dengan menghitung banyaknya kemunculan term t_i dalam dokumen d_j.

$$idf_i = \log\left(\frac{N}{df_i}\right) \quad (2)$$

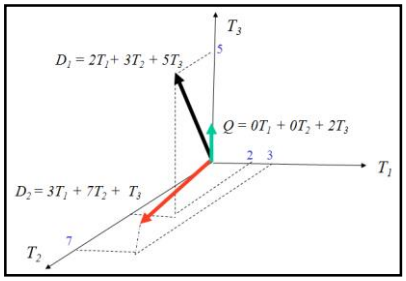
Dengan idf_i adalah *inverse document frequency*, *N* adalah jumlah dokumen yang terambil oleh sistem, dan df_i adalah banyaknya dokumen dalam koleksi dimana term t_i muncul di dalamnya, maka perhitungan idf_i digunakan untuk mengetahui banyaknya *term* yang dicari (df_i) yang muncul dalam dokumen lain yang ada pada *database*.

Perhitungan bobot dari *term* tertentu dalam sebuah dokumen menggunakan perkalian nilai *tf* dan *idf* menunjukkan bahwa deskripsi terbaik dari dokumen adalah *term* yang banyak muncul dalam dokumen tersebut dan sangat sedikit muncul pada dokumen yang lain. Perhitungan bobot *term frequency-inverse document frequency* adalah sebagai berikut [20].

$$W_{i,j} = tf_{i,j} log \frac{N}{df_i} \quad (3)$$

D. Vector Space Model

Vector Space Model (VSM) merupakan suatu metode yang digunakan untuk mengukur tingkat kedekatan atau kesamaan (*similarity*) *term* dengan cara pembobotan pada *term* [21]. Dokumen diasumsikan sebagai sebuah vektor-vektor yang memiliki jarak (*magnitude*) dan arah (*direction*). Dalam metode ini, sebuah *term* direpresentasikan dengan sebuah dimensi dari ruang vektor. *Term* yang digunakan umumnya berdasarkan kepada *term* yag ada pada *query* atau *keyword*. Relevansi sebuah dokumen ke sebuah *query* didasarkan pada similaritas di antara vektor dokumen dan vektor *query* [20]. Perhitungan kesamaan antara vektor *query* dengan vektor dokumen dilihat dari sudut yang paling kecil. Gambar 1 merupakan contoh dari model ruang vektor tiga dimensi untuk 2 dokumen di mana D adalah dokumen, Q adalah *query*, dan T adalah *term* yang menjadi dimensi dari VSM



Gambar 1. Vector Space Model [21]

Dalam *Vector Space Model*, koleksi dokumen direpresentasikan sebagai sebuah matriks *term document* (matriks *term frequency*). Setiap sel dalam matriks bersesuaian dengan bobot yang diberikan dari suatu *term* dalam dokumen yang ditentukan. Nilai nol menunjukkan *term* tersebut tidak ada dalam dokumen. Gambar 2 menunjukkan matriks *term document* dengan *n* dokumen dan *t term* [21].

	T_1	T_2	T_3	$T_{...}$	T_t
D_1	W_{11}	W_{21}	W_{31}	$\square_{...}$	T_{t1}
D_2	W_{12}	W_{22}	W_{32}	$\square_{...}$	T_{t2}
D_3	W_{13}	W_{23}	W_{33}	$\square_{...}$	T_{t3}
$D_{...}$	$\square_{...}$	$\square_{...}$	$\square_{...}$	$\square_{...}$	$\square_{\square}$
D_n	W_{1n}	W_{2n}	W_{3n}	$\square_{...}$	T_{tn}

Gambar 2 Matriks Term Document

E. Cosine Similarity

Cosine similarity merupakan metode pengukuran tingkat kemiripan yang didapatkan dari perbandingan hasil perkalian *cosine angle* 2 buah vektor. *Cosine* 0 adalah 1 dan kurang dari 1 terhadap nilai *angle* yang lain maka 2 buah vektor dikatakan mirip ketika nilai *cosine similarity* adalah 1 [22]. Perhitungan *cosine similarity* ditunjukkan oleh persamaan berikut.

$$Sim\,(q, d_j) = \frac{q \times d_j}{|q| \times |d_j|} = \frac{\Sigma_{i=1}^{n} W_{i,q} \times W_{i,j}}{\sqrt{\Sigma_{i=1}^{t} (W_{i,q})^2} \times \sqrt{\Sigma_{i=1}^{t} (W_{i,j})^2}} \quad (4)$$

Dengan $Sim(q,d_j)$ adalah similaritas antara *query* dan dokumen. |q| adalah jarak *query*. $|d_j|$ adalah jarak dokumen. $W_{i,j}$ adalah bobot dokumen ke-*i*. $W_{i,q}$ adalah bobot *query* dokumen ke-*i*. Similaritas antara *query* dan dokumen atau berbanding lurus terhadap hasil perkalian jumlah bobot *query* (*q*) dengan bobot dokumen (d_j) dan berbanding terbalik terhadap perkalian dari akar jumlah kuadrat *q* (|q|) dengan akar jumlah kuadrat dokumen $|d_j|$ [21]. Perhitungan similaritas menghasilkan bobot dokumen yang mendekati nilai 1 atau menghasilkan bobot dokumen yang lebih besar dibandingkan dengan nilai yang dihasilkan dari perhitungan *inner product* [21].

F. Precision, Recall, dan F-Measure

Dalam penilaian relevansi ada dua hal penting yang umumnya digunakan dalam mengukur kemampuan suatu sistem dalam memanggil dokumen sesuai dengan istilah yang diformulasikan, yaitu *precision* dan *recall* [23]. Penggunaan perhitungan *precision* dan *recall* dapat menggunakan Tabel 1.

Tabel 1. Perhitungan Precision dan Recall [34]

	Relevant	Not Relevant	Total
Retrieved	A	B	A + B
Not Retrieved	C	D	C + D
Total	A + C	B + D	

Recall adalah perbandingan antara jumlah *records* relevan yang didapatkan dengan jumlah keseluruhan *records* relevan pada *database*.

$$R = \frac{Number\ of\ relevant\ items\ retrieved}{Total\ number\ of\ relevant\ items\ in\ collection} \quad (5)$$

Precision adalah perbandingan antara jumlah *records* relevan yang didapatkan dengan jumlah keseluruhan *records* yang relevan maupun yang tidak relevan.

$$P = \frac{Number\ of\ relevant\ items\ retrieved}{Total\ number\ of\ items\ retrieved} \quad (6)$$

Recall sebenarnya sulit untuk diukur karena jumlah seluruh dokumen yang relevan dalam *database* sangat besar [23]. Oleh karena itu, *precision* yang dijadikan salah satu ukuran yang digunakan untuk menilai keefektifan suatu sistem pencarian. F-*Measure* sering disebut F1 *score* adalah *harmonic mean* atau nilai rataan harmonis antara perhitungan *precision* dan *recall* [24]. F-*Measure* dapat dihitung menggunakan persamaan berikut [24].

$$F1 = 2 * \frac{precision * recall}{precision + recall} \quad (7)$$

III. PERANCANGAN SISTEM

A. Model Aplikasi

Gambar 3 menunjukkan model dari aplikasi rekomendasi buku. Terdapat 5 bagian, yaitu modul untuk mendapatkan data deskripsi buku, modul yang menjalankan preprocessing data, modul perhitungan nilai kemiripan antar buku, modul untuk mendapatkan rekomendasi, dan user interface aplikasi. Berikut penjelasan setiap bagian.

1. Books' Description Module

   Modul yang berfungsi untuk memperoleh data deskripsi buku. Data deskripsi buku didapatkan dengan menggunakan Google Books API atau Goodreads API.

2. Preprocessing Data Module

   Modul yang berfungsi untuk melakukan proses *preprocessing* data, di mana terdapat proses *case folding*, *tokenization*, *filtering*, dan *stemming* terhadap data deskripsi buku.

3. Calculating Books Similarity Score Module

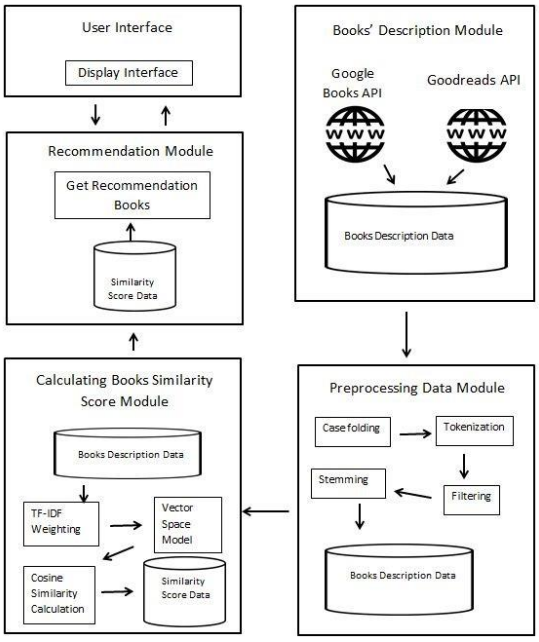
   Modul yang berfungsi untuk menghitung skor kemiripan antar deskripsi buku dengan menggunakan algoritma TF-IDF dalam memberikan bobot setiap *term* pada deskripsi, lalu menggunakan Vector Space Model dalam menggambarkan vektor dokumen yang ada. Kemudian, kemiripan antar deskripsi buku dihitung menggunakan rumus *cosine similarity*.

4. Recommendation Module

   Modul yang berfungsi untuk mengambil data rekomendasi buku yang tersimpan dalam database untuk ditampilkan melalui *user interface*.

5. User Interface

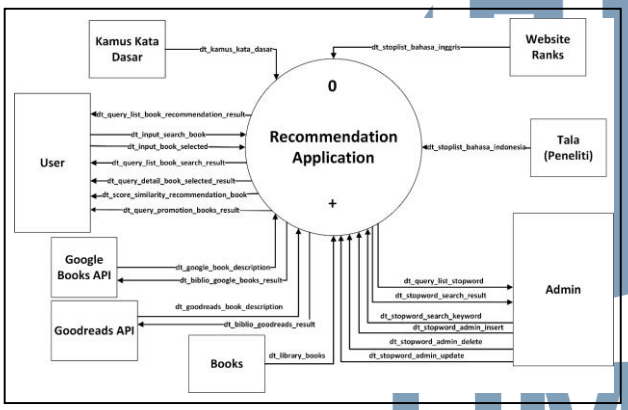
   Bagian di mana pengguna berinteraksi dengan aplikasi rekomendasi. Pengguna dapat memilih buku secara spesifik dan akan ditampilkan rekomendasi buku berdasarkan buku yang dipilih tersebut.



Gambar 3. Model Aplikasi Rekomendasi Buku

B. Data Flow Diagram

Alur perpindahan data yang ada dalam aplikasi dijabarkan menggunakan *Data Flow Diagram* (DFD). Gambar 4 merupakan *context diagram* yang menunjukkan garis besar perpindahan data yang terjadi pada aplikasi. *Context diagram* merupakan DFD level 0. Dalam context diagram ini, terdapat satu proses utama yaitu Recommendation Application dan 8 entitas, yaitu *website ranks*, Tala (peneliti), kamus kata dasar, *books*, Google Books API, Goodreads API, *admin*, dan *user*. Aplikasi rekomendasi ini memiliki aktivitas yang terjadi pada bagian *front-end* dan *back-end*. Entitas *user* memiliki 2 data keluar dan 5 data masuk menuju proses Recommendation Application. Tujuh buah data ini merupakan data yang mewakili perpindahan data yang terjadi pada bagian *front-end*. Entitas *website ranks*, Tala (Peneliti), *books*, dan kamus kata dasar memiliki masing-masing 1 data keluar menuju proses. Entitas Google Books API dan Goodreads API memiliki masing-masing 1 data keluar dan 1 data masuk. Entitas *admin* memiliki 4 data keluar dan 2 data masuk.



Gambar 4. Context Diagram Aplikasi Rekomendasi

C. *Flowchart*

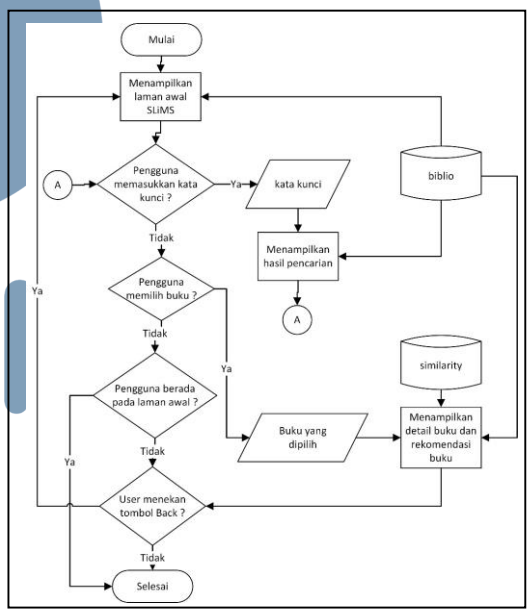
1. Flowchart Sistem Pencarian SLiMS

Gambar 5 merupakan *flowchart* gambaran sistem pencarian buku SLiMS secara umum. Proses diawali dengan menampilkan laman awal SLiMS yang menampilkan buku-buku promosi dari perpustakaan UMN sehingga data buku dari tabel biblio dibutuhkan. Selanjutnya, apakah pengguna ingin melakukan pencarian buku dengan keyword tertentu, jika benar pengguna akan memasukkan keyword pencarian, dan SLiMS akan menampilkan daftar hasil pencarian buku yang sesuai dengan *keyword* tersebut. Setelah mendapatkan hasil pencarian, apakah pengguna memilih salah satu buku, jika benar sistem SLiMS akan menampilkan laman detail buku tersebut beserta rekomendasinya.

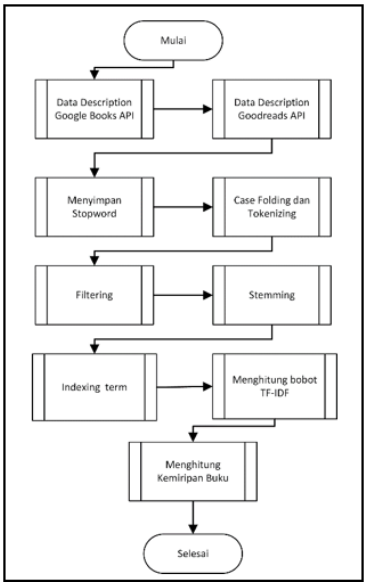
Pada laman detail buku, terdapat tombol Back yang berfungsi untuk berpindah laman menuju laman sebelumnya, yaitu laman daftar pencarian buku. Apakah pengguna melakukan klik pada tombol Back tersebut, jika benar, pengguna akan dibawa kembali menuju laman daftar pencarian buku, tetapi jika tidak, pengguna akan tetap berada pada laman detail buku. Kondisi yang telah dijelaskan sebelumnya menggambarkan alur sistem jika pengguna melakukan pencarian buku saat berada di laman awal SLiMS. Namun, jika setelah dari laman awal SLiMS, pengguna memilih buku yang dipromosikan oleh perpustakaan UMN, sistem SLiMS akan menampilkan laman detail dari buku yang dipilih pengguna tersebut.

2. Flowchart Back End Skor Kemiripan Antar Buku

Gambar 6 merupakan *flowchart back-end* yang menjelaskan alur proses dalam mendapatkan skor kemiripan antar buku. Proses diawali dengan mendapatkan deskripsi dari Google Books API, dan Goodreads API. Setelah deskripsi didapatkan, *stopword* yang berfungsi untuk membantu tahap *filtering* perlu disimpan. *Preprocessing* data dimulai dengan berjalannya tahap tokenizing, dan dilanjutkan dengan tahap *case folding*, *filtering*, dan *stemming*. *Term* yang terdapat pada deskripsi disimpan dalam *database*. Lalu, *term* yang ada pada deskripsi buku, dihitung bobot TF-IDF-nya. Tahap terakhir yang perlu dijalankan adalah menghitung kemiripan antar buku.



Gambar 5. Flowchart Sistem SLiMS



Gambar 6. Flowchart Back End Aplikasi

IV. IMPLEMENTASI DAN UJI COBA

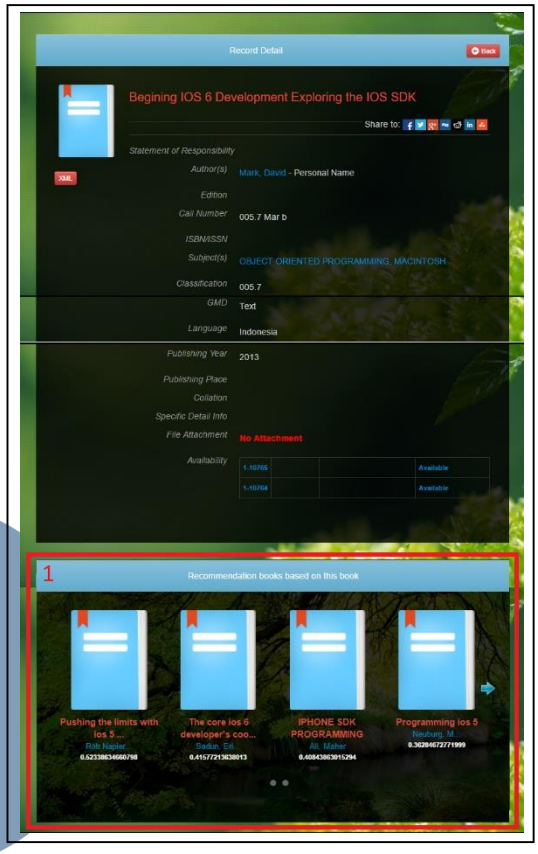
A. *Implementasi*

Dalam membangun aplikasi rekomendasi buku ini, digunakan bahasa pemrograman PHP, dilengkapi dengan *database* MySQL, dan berjalan pada *web server* Apache.

Sebelum implementasi *Vector Space Model* dilakukan, diperlukan data berupa deskripsi buku yang diperoleh dari Goodreads dan Google Books menggunakan API. Jumlah koleksi yang dimiliki oleh perpustakaan UMN adalah 17.114 judul (Maret 2017). Data laporan magang dan skripsi tidak digunakan dalam penelitian ini karena aplikasi rekomendasi difokuskan untuk buku. Setelah dilakukan penyaringan data terhadap laporan magang dan skripsi, didapatkan koleksi berjumlah 9.804 judul.

Aplikasi rekomendasi buku diimplementasikan dengan dua bagian, yaitu bagian *back-end* dan bagian *front-end*. Bagian *back-end* dibangun terlebih dahulu karena aplikasi yang dibangun memiliki data utama yang perlu disiapkan terlebih dahulu sebelum dapat ditampilkan kepada pengguna. Bagian *back-end* digunakan untuk menjalankan pengumpulan data deskripsi buku, *preprocessing* data, perhitungan bobot *term* deskripsi yang ada menggunakan TF-IDF, dan perhitungan kemiripan antar buku dengan menggunakan *cosine similarity*. Selain itu, terdapat satu laman *web* yang dapat digunakan oleh pengguna (*admin*) untuk menambahkan stopword pada stoplist yang tersedia. *Stoplist* digunakan pada tahap *filtering* pada *preprocessing* data.

Gambar 7 menunjukkan hasil implementasi aplikasi rekomendasi buku. Rekomendasi buku ditunjukkan pada bagian 1. Selain data detail buku, laman ini juga menampilkan buku rekomendasi beserta informasi skor kemiripan yang dimiliki buku tersebut dengan buku yang telah dipilih secara spesifik.



Gambar 7. Tampilan Aplikasi Rekomendasi Buku

B. *Skenario Uji Coba Aplikasi*

Buku-buku yang terdapat pada perpustakaan UMN terdiri dari buku berbahasa Inggris dan buku berbahasa Indonesia. Oleh karena itu, pengujian dilakukan sebanyak dua kali. Pengujian pertama dilakukan dengan menggunakan sampel buku yang relevansinya merupakan gabungan dari kedua bahasa tersebut. Sebagai contoh, buku berjudul "Accounting Information Systems 13th" memiliki relevansi dengan buku berjudul "Sistem Informasi Akuntansi", dan "Accounting Information Systems and Internal Control (2nd edition)".

Berbeda dengan pengujian pertama, sampel buku pada pengujian kedua menggunakan sampel buku yang relevansinya sesuai dengan bahasa yang digunakan. Sebagai contoh, buku berjudul "Accounting Information Systems 13th" memiliki relevansi dengan buku berjudul "Accounting Information Systems and Internal Control (2nd Edition)", tetapi tidak memiliki relevansi dengan buku berjudul "Sistem Informasi Akuntansi" karena deskripsi buku tersebut menggunakan bahasa yang berbeda dengan deskripsi buku "Accounting Information Systems 13th".

1. Pengujian Sampel Buku dengan Dua Bahasa

Pengujian dalam penelitian ini mengambil sampel berjumlah 158 buku dengan 10 buku acuan yang merupakan buku yang paling sering dipinjam di perpustakaan UMN. Data buku ini dapat diperoleh dengan mengakses SLiMS UMN pada link http://slims.umn.ac.id/index.php?p=statistics. Sampel ini dipilih karena latar belakang penelitian yang berusaha memberikan alternatif informasi buku lain ketika buku yang hendak dipinjam sedang dipinjam. Tingginya angka peminjaman membuat probabilitas ketersediaan buku-buku tersebut semakin kecil. Dengan demikian, hasil pengujian yang dilakukan juga dapat menjawab apakah imlementasi metode *Content-based Filtering* pada aplikasi rekomendasi buku menggunakan *Vector Space Model* telah memberikan alternatif informasi buku yang relevan dengan baik.

Tabel 2 menunjukkan rata-rata nilai precision dan recall pada setiap batas ambang dari buku sampel yang telah ditentukan.

Tabel 2. Rata-rata Precision and Recall pada Pengujian 2 Bahasa

Threshold	Precision	Recall	F-Measure
0,7	0,1	0,003	0,005
0,6	0,2	0,016	0,029
0,5	0,45	0,053	0,095
0,4	0,285	0,062	0,102
0,3	0,26	0,085	0,129
0,2	0,486	0,208	0,292
0,1	0,415	0,431	0,423
0	0,279	0,519	0,363

Dengan demikian, mengacu pada Tabel 2, nilai F-*Measure* tertinggi dicapai dengan batas ambang 0,1, yaitu sebesar 42,3% dengan nilai *precision* sebesar 41,5%, dan nilai *recall* sebesar 43,1%.

2. Pengujian Sampel Buku dengan Satu Bahasa

Pengujian kedua mengambil sampel berjumlah 121 buku dengan 10 buku acuan yang sama dengan pengujian pertama. Berbeda dengan pengujian pertama, pada pengujian kedua, topik utama yang digunakan untuk mencari buku yang relevan tidak diterjemahkan ke dalam dua bahasa, melainkan hanya menggunakan bahasa yang digunakan pada deskripsi buku acuan. Sebagai contoh, buku berjudul "Accounting Information Systems 13th" memiliki relevansi dengan buku berjudul "Accounting Information Systems and Internal Control (2nd edition)", tetapi tidak relevan dengan "Sistem Informasi Akuntansi".

Tabel 3 menunjukkan rata-rata nilai *precision* dan *recall* pada setiap batas ambang dari buku sampel yang telah ditentukan.

Tabel 3. Rata-rata Precision and Recall pada Pengujian 1 Bahasa

Threshold	Precision	Recall	F-Measure
0,7	0,1	0,005	0,01
0,6	0,2	0,028	0,049
0,5	0,50	0,096	0,161
0,4	0,400	0,111	0,174
0,3	0,376	0,150	0,214
0,2	0,508	0,350	0,415
0,1	0,465	0,674	0,550
0	0,319	0,768	0,451

Dengan demikian, mengacu pada Tabel 3, nilai F-*Measure* tertinggi dicapai dengan batas ambang 0,1, yaitu sebesar 55% dengan nilai *precision* sebesar 46,5%, dan nilai *recall* sebesar 67,4%.

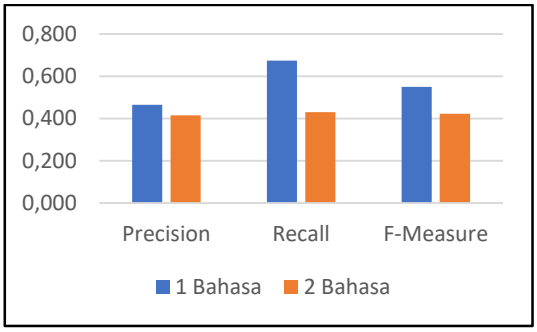
C. Analisis Hasil Pengujian

Dari hasil kedua pengujian, diketahui bahwa nilai F-*Measure* yang diperoleh dari pengujian yang menggunakan 2 bahasa sebagai sampel pengujian, yaitu sebesar 42,3% pada batas ambang 0,1. Namun, ketika menggunakan 1 bahasa sebagai sampel pengujian, nilai F-*Measure* yang diperoleh yaitu sebesar 55% pada batas ambang yang sama. Dengan demikian, pengujian menggunakan 1 bahasa memiliki nilai F-*Measure* yang lebih tinggi sebesar 12,7% dibandingkan dengan pengujian 2 bahasa.

Selain itu, nilai *precision* pada pengujian yang menggunakan relevansi 2 bahasa, yaitu sebesar 41,5%, sedangkan pada pengujian yang menggunakan relevansi 1 bahasa, nilai *precision* yang didapat yaitu sebesar 46,5% sehingga pengujian menggunakan 1 bahasa sebagai sampel pengujian memiliki nilai *precision* lebih tinggi sebesar 5% dari pengujian menggunakan 2 bahasa sebagai sampel pengujian.

Nilai *recall* pada pengujian menggunakan 1 bahasa sebagai sampel pengujian juga lebih tinggi dengan selisih sebesar 24,3%, di mana pada pengujian menggunakan 2 bahasa sebagai sampel pengujian, nilai *recall* yang dihasilkan yaitu sebesar 43,1% pada batas ambang 0,1, sedangkan pada pengujian menggunakan 1 bahasa sebagai sampel pengujian, nilai *recall* yang dihasilkan yaitu sebesar 67,4% pada batas ambang yang sama.

Gambar 8 merupakan grafik yang menggambarkan perbedaan skor *precision*, *recall*, dan F-*Measure* pada batas ambang 0,1 dari hasil kedua pengujian. Semakin tinggi skor *precision*, *recall*, dan F-*Measure* yang dihasilkan maka semakin baik pula aplikasi rekomendasi yang dibuat.



Gambar 8. Grafik Perbandingan Pengujian

Dari Gambar 8 terlihat bahwa skor *precision*, *recall*, dan F-*Measure* pada pengujian menggunakan 1 bahasa lebih tinggi dari pengujian menggunakan 2 bahasa. Dengan demikian, perbedaan bahasa yang digunakan berpengaruh terhadap jumlah buku relevan yang terpanggil oleh aplikasi.

Selain itu, deskripsi yang dimiliki buku juga dapat mempengaruhi terpanggilnya koleksi lain yang dinilai tidak relevan. Sebagai contoh, dari hasil implementasi Vector Space Model yang telah dibuat, didapatkan bahwa buku "A First Look at Communication Theory" memiliki kemiripan dengan buku "Advanced Accounting (5ed)". Padahal, kedua buku tersebut tidak revelan. Namun, karena terdapat kata "approach", "balanc", dan "reflect" pada hasil proses stemming deskripsi kedua buku tersebut, kedua buku tersebut dinilai memiliki kemiripan. Kata "approach", "balanc", dan "reflect" sendiri bukan merupakan *stopword* sehingga pada proses *filtering*, ketiga kata tersebut tidak dihapus. Oleh karena itu, selain faktor bahasa yang digunakan, deskripsi buku juga mempengaruhi nilai *precision*, *recall*, dan F-*Measure* yang dihasilkan oleh aplikasi.

V. SIMPULAN

Metode *Content-based Filtering* pada aplikasi rekomendasi buku telah berhasil diimplementasikan dengan menggunakan *Vector Space Model*. Aplikasi rekomendasi buku dapat menampilkan buku-buku yang relevan dengan buku yang dipilih oleh pengguna berdasarkan kemiripan deskripsi yang dimiliki setiap buku. Berdasarkan proses pengujian yang telah dilakukan dengan sampel relevansi 1 bahasa dan nilai batas ambang cosine similarity 0,1, diperoleh nilai F-Measure sebesar 55%. Nilai tersebut lebih tinggi jika dibandingkan dengan nilai F-Measure untuk pengujian dengan sampel relevansi 2 bahasa. Deskripsi buku yang tersedia dan perbedaan bahasa yang digunakan menjadi beberapa faktor yang mempengaruhi nilai F-Measure yang diperoleh.

Aplikasi rekomendasi buku yang dibuat masih memiliki kekurangan dan keterbatasan. Saran untuk pengembangan aplikasi rekomendasi buku ini adalah sebagai berikut.

1. Dalam penelitian selanjutnya, deskripsi buku yang menjadi data aplikasi rekomendasi dapat diterjemahkan menjadi satu bahasa sehingga buku-buku yang berbeda bahasa, tetapi tetap relevan dapat tetap terpanggil oleh aplikasi.
2. Mengombinasikan *Vector Space Model* dengan algoritma *rocchio* untuk meningkatkan nilai rata-rata *precision*.
3. Mengembangkan sistem rekomendasi berbasis semantik sehingga deskripsi pada buku tidak hanya dianalisis pada tahap *lexical*, tetapi juga dianalisis hingga tahap semantik.

DAFTAR PUSTAKA

[1] Wandi, N., Hendrawan, R. A., & Mukhlason, A. (2012). Pengembangan Sistem Rekomendasi Penelusuran Buku dengan Penggalian Association Rule Menggunakan Algoritma Apriori (Studi Kasus Badan Perpustakaan dan Kearsipan Provinsi Jawa Timur). Jurnal Teknik ITS Vol. 1, 445-449.

[2] Rani, P., Solanki, P., Puntambekar, V., Borse, V., & Vaidya, M. A. (2014). LIBRS: Library Recommendation System Using Hybrid Filtering. International Journal of Technical Research and Application, 78-81.

[3] Universitas Multimedia Nusantara. (2017). Universitas Multimedia Nusantara Online Library Profile Page. Diambil kembali dari Universitas Multimedia Nusantara Online Library: http://library.umn.ac.id/umnlibrary/umnlibrary/content/Profile

[4] SLiMS. (2017). SLiMS Users Forum. Diambil kembali dari SLiMS Users Forum: https://forum.slims.web.id.

[5] Miraldi, R. N. (2013). Implementasi Algoritma FP-Growth untuk Sistem Rekomendasi Buku di Perpustakaan UKDW. Yogyakarta: Universitas Kristen Duta Wacana.

Ghadling, P. S., Belavadi, K., Bhegade, S., Ghojage, P., & Kamble, S. (2015). Digital Library: Using Hybrid Book Recommendation Engine. International Journal Of Engineering And Computer Science, 1-3.

[6] Pazzani, M. J., & Billsus, D. (2007). Content-based Recommendation Systems. Dalam P. Brusilovsky, A. Kobsa, & W. Nejdl, The Adaptive Web (hal. 325-341). Palo Alto: Springer Berlin Heidelberg.

[7] Oktoria, R., Maharani, W., & Firdaus, Y. (2010). Content Based Recommender System Menggunakan Algoritma Apriori . Konferensi Nasional Sistem dan Informatika , 125-129.

[8] Lehinevych, T., Kokkinis-Ntrenis, N., & Siantikos, G. (2014). Discovering Similarities for Content-Based Recommendation and Browsing in Multimedia Collections. Tenth International Conference on Signal-Image Technology & Internet-Based Systems, 237-243.

[9] Zhang, H., Li, J., Ji, Y., & Ye, Y. (2015). Content-based Movie Recommending Using a Triple Wing Harmonium Model. IEEE, 1096-1101.

[10] Leskovech, J., Rajaraman, A., Ullman, J.D. (2010). Mining of Massive Datasets. New York: Cambridge University Press.

[11] Husni. (2010). Aplikasi Pencarian Karya Tulis Ilmiah Berbasis Web Menggunakan Sistem Rekomendasi. *Rekayasa Jurnal Vol 3 No. 1*, 36-41.

[12] Han, J., & Kamber, M. (2000). Data mining: Concepts and Techniques. New York: Morgan-Kaufman.

[13] Tan, A. (1999). Text Mining: The state of the art and the challenges. In Proc of the Pacific Asia Conf on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge Discovery from Advanced Databases.

[14] [Handayani, N. M. (2014). Perancangan dan Implementasi Sistem Rekomendasi Pencarian Buku Perpustakaan

Menggunakan Metode Vector Space Model (Studi Kasus Perpustakaan Universitas Udayana). Universitas Udayana.

[15] Robertson, S. (2004). Understanding Inverse Document Frequency: On theoretical arguments for IDF, England : Journal of Documentation, Vol. 60, 502–520

[16] Satya, K. P., & Murthy, J. V. (2012). Clustering Based on Cosine Similarity Measure. 508-512.

[17] Anggiharto, A. (2013). Implementasi Algoritma TF-IDF dan Vector Space Model Untuk Klasifikasi E-Book Berbasis Library of Congress. Tangerang: Universitas Multimedia Nusantara.

[18] Uden & V., M. (2011). Rocchio: Relevance Feedback in Learning Classification Algorithms. Department of Computing Science University of Nijmegen.

[19] Yates, B. & Neto, R. (1999). Chapter 3. Dalam Baeza-Yates & Ribeiro-Neto, Modern Information Retrieval. New York: ACM Press.

[20] Amin, F. (2012). Sistem Temu Kembali Informasi dengan Metode Vector Space Model. Jurnal Sistem Informasi Bisnis, 80.

[21] Saputra, W. S. & Muttaqin, F. (2013). Pengenalan Karakter Pada Proses Digitalisasi. 51-56.

[22] Hasugian, J. (2006). Penggunaan Bahasa Alamiah dan Kosa Kata Terkontrol Dalam Sistem Temu Kembali Informasi Berbasis Teks. Jurnal Pustaka: Jurnal Studi Perpustakaan dan Informasi.

[23] Hripsack, G. & Rothschild A.S. (2005). Technical Brief: Agreement, the F-Measure, and Reliability in Information Retrieval. J Am Med Inform Assoc, 12:296-29.