BAB II
TINJAUAN PUSTAKA

2.1 Kajian Terkait

Penelitian ini mengkorelasikan beberapa penelitian yang telah dilakukan sebelumnya terkait dengan konsep rekomendasi masker wajah tradisional, metode *Cosine Similarity*, dan TF-IDF. Selanjutnya menjelaskan beberapa penelitian terkait dengan ranah penelitian tersebut.

Naf'an et al (2019) melakukan penelitian terkait kemiripan dokumen teks bertujuan untuk mendeteksi kemiripan dokumen teks menggunakan *Cosine Similarity* dan pembobotan TF-IDF sehingga dapat menentukan nilai plagiarisme. Penelitian tersebut menggunakan dokumen abstrak bahasa Indonesia dalam melakukan perbandingan teks. Hasilnya menunjukkan bahwa saat dilakukan *stemming* nilai kemiripan lebih tinggi rata-rata 10% daripada tidak dilakukan proses *stemming*. Penelitian tersebut menghasilkan nilai similaritas di atas 50% untuk dokumen yang tingkat kemiripannya tinggi. Sedangkan untuk dokumen dengan tingkat kemiripan rendah atau tidak berplagiat menghasilkan nilai *similarity* di bawah 40%.

Fikrina et al (2017) melakukan penelitian terhadap klasterisasi teks berbahasa Indonesia menggunakan TF-IDF dan kesamaan kosinus yang menghasilkan kesimpulan bahwa evaluasi hasil klasterisasi dilakukan dengan menghitung nilai *silhouette coefficient* dari hasil klasterisasi dokumen. Berdasarkan percobaan yang dilakukan, didapatkan bahwa perhitungan kesamaan kosinus tidak selalu meningkatkan hasil klasterisasi dokumen. Pada beberapa parameter pemilihan fitur, penggunaan TF-IDF tanpa kesamaan kosinus memiliki hasil klasterisasi yang lebih baik. Nilai batas maksimal dan minimal dokumen frekuensi (*df*) berpengaruh pada pencarian jumlah klaster terbaik. Dari hasil percobaan didapatkan jumlah klaster terbaik pada nilai maksimum *df* 0,8 dan nilai minimum *df* 0,3. Jumlah klaster terbaik sudah mencapai kelompok *medium structure* dengan nilai *silhouette coefficient* 0,604287007 untuk proses klasterisasi dengan fitur

pembobotan TF-IDF dan 0,618543225 untuk proses klasterisasi dengan kombinasi TF-IDF dan perhitungan kesamaan kosinus.

Pandalu (2014) melakukan penelitian dengan menggunakan metode *text mining* yang mengimplementasikan *Cosine Similarity* untuk peringkatan jenis obat yang paling mirip. Hal ini diperlukan karena terdapat berbagai macam jenis obat untuk jenis penyakit yang sama. Pengguna memasukkan gejala penyakit yang diderita, kemudian *system* akan mengukur kemiripan gejala yang diberikan dengan manfaat obat tradisional.

Aminudin et al (2020) melakukan penelitian terhadap kinerja pencarian barang hilang "*Lost And Found*" Pada Kampus 3 Universitas Muhammadiyah Malang. Penelitian tersebut mampu melakukan proses pencarian barang dengan menggunakan kata kunci barang serta lokasi kehilangan atau penemuan barang menggunakan metode *Cosine Similarity* sehingga menghasilkan kesimpulan bahwa informasi yang didapatkan dapat lebih cepat dan akurat. Berdasarkan pengujian kepada 100 responden secara acak, didapatkan hasil sebesar 77,8% dan dapat dikatakan bahwa *website* yang sudah dibangun ini sudah diterima atau layak bagi pengguna.

Nurdiana et al (2016) melakukan penelitian terhadap informasi sistem temu kembali yang digunakan untuk mencari informasi yang relevan dengan kebutuhan pengguna secara otomatis berdasarkan kesesuaian dengan *query* dari kumpulan informasi. Pada sistem ini terdapat beberapa algoritma yang digunakan untuk menentukan kesamaan (derajat kesamaan) atau algoritma kesamaan yang relevan, cosinus, *Jaccard*, dan tetangga terdekat (*K-NN*) untuk membandingkan yang lebih relevan dengan aplikasi terjemahan alquran. Hasil pengujian membuktikan bahwa persamaan cosinus memiliki nilai tertinggi dengan persentase 41% dibandingkan dengan *Jaccard* 19% algoritma dan tetangga terdekat (*K-NN*) 40% terjemahan AL-Quran sebanyak 6326 dokumen dan 33 kueri eksperimen yang berbeda.

Aisyana (2017) melakukan penelitian terhadap sebuah *system* pencarian yang dapat menemukan resensi buku yang relevan terhadap *query* menggunakan

Cosine Similarity sehingga *user* mendapat gambaran informasi yang dikandung sebuah buku. Peneliti menggunakan *WordNet* Bahasa untuk mengukur keterkaitan makna antar kata. Peneliti berharap Sistem pencari resensi buku dapat mempermudah *user* dalam mencari resensi buku yang diinginkan. Serta *user* dapat memahami informasi yang diberikan resensi buku.

Hartanto (2019) melakukan penelitian terhadap aplikasi pencarian ensiklopedia kebidanan menggunakan TF-IDF dan *Cosine Similarity*. TF-IDF dan *Cosine Similarity* dapat digunakan untuk melakukan pembobotan data dan kemiripan dokumen hasil pencarian secara langsung. Hasil pencarian yang ditampilkan dapat mewakili keseluruhan topik kebidanan dari halaman hasil pencarian. Solusi permasalahan dalam penelitian ini yaitu pengguna dapat langsung ditampilkan dalam waktu yang cepat. Untuk menguji hasil *recall* dan perhitungan presisi dari 813 dokumen data dalam *database*, sepuluh kata kunci pencarian yang berbeda digunakan untuk melihat perbandingan hasil, menghasilkan *recall* 6,5 dan presisi 52,7.

Anshori (2017) melakukan penelitian terhadap sebuah model pencarian hadits yang relevan menggunakan metode *Cosine Similarity*. Uji coba yang dilakukan peneliti adalah pengukuran kemiripan dan penentuan *threshold*. Hasil yang didapat model berhasil menghitung kemiripan dengan jangkauan nilai *cosine* antara 0,05-0,25, semakin besar maka semakin mirip esai jawaban tersebut dengan kunci jawaban soal. *Threshold* terbaik yang diperoleh adalah 0,1 dengan *recall* 83% dan *precision* 46.

2.2 Masker Wajah Tradisional

Menurut Fujiko (2022) masker wajah tradisional merupakan masker berbahan alam yang bebas dari bahan kimia. Masker wajah tradisional memiliki banyak manfaat yang tentunya tidak kalah dari perawatan di dokter kecantikan. Masker wajah adalah masker kecantikan yang berwujud sediaan gel, pasta dan serbuk yang dioleskan untuk membersihkan dan mengencangkan kulit terutama

kulit wajah. Masker wajah juga berfungsi sebagai pembawa bahan-bahan aktif yang berguna bagi kesehatan kulit, seperti ekstra tumbuhan, minyak esensial, atau rumput laut yang dapat diserap oleh permukaan kulit untuk dibawa kedalam sirkulasi.

2.3 *Information Retrieval*

Menurut Safitri (2013) *Information Retrieval* (IR) merupakan bagian dari *computer science* yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dari dokumen-dokumen itu sendiri. *Information Retrieval* merupakan suatu pencarian informasi (biasanya berupa dokumen) yang didasarkan pada suatu masukkan pengguna yang diharapkan dapat memenuhi keinginan pengguna dari kumpulan dokumen yang ada.

2.4 *Preprocessing*

Menurut Feldman & Sanger (2009) *preprocessing* merupakan proses perubahan bentuk data yang terstruktur sembarang menjadi data yang terstruktur sesuai kebutuhan untuk proses dalam *text mining*. Menurut Kadhim (2018) *preprocessing* adalah tahapan mereduksi sebuah kalimat atau beberapa kata menjadi satu bentuk kata tunggal. Tujuan utama dari *preprocessing* teks adalah untuk mendapatkan suatu bentuk kata tunggal (*root word*) dari sebuah dokumen, dengan menghapus fitur non-informatif seperti (kata sambung, imbuhan, angka dan karakter khusus).

2.5 *Case Folding*

Menurut Terán & Mancera (2019) *case folding* adalah metode untuk mengubah semua huruf dalam *dataset* menjadi huruf kapital atau semua kecil. Hal ini dilakukan untuk memudahkan proses analisis *dataset* dan mengurangi jumlah memori penggunaan. Menurut Mustaqim et al (2020) contoh *case folding* adalah mengubah frasa "Hutan dalam KEBAKARAN BESAR di Kalimantan!" Untuk "karena hutan sedang kebakaran hebat di kalimantan!". *Case folding* membantu lemmatisasi dan *stemming* proses untuk menemukan kecocokan untuk setiap data dalam kamus. Menurut Benbrahim & Bramer (2009) karakter-karakter selain huruf

‘a’ sampai ‘z’ (tanda baca dan angka) akan dihilangkan dari data dan dianggap sebagai delimiter.

2.6 *Tokenizing*

Menurut Rezki et al (2020) *tokenizing* adalah tahap pemotongan atau pembagian *string* menjadi urutan token berdasarkan tiap kata yang menyusunnya, atau juga disebut proses pemecah sekumpulan kalimat menjadi satuan kata. Strategi umum yang dilakukan pada tahap *tokenizing* adalah memotong kata pada *white space* atau spasi dan membuang karakter tanda baca. Tahap *tokenizing* membagi urutan karakter menjadi kalimat dan kalimat menjadi token.

2.7 *Stopword/Filtering*

Menurut Rahutomo & Ririd (2018) salah satu langkah pemrosesan teks di bidang sistem temu kembali informasi teks atau *text mining* adalah pembersihan teks dari kata-kata yang tidak relevan dijadikan indeks. Di dalam sebuah dokumen teks bisa jadi terdapat banyak jenis kata seperti kata depan, kata sambung, kata ganti, kata sifat, dan lain sebagainya. Sebagian kata tersebut bisa jadi tidak berpotensi dijadikan indeks dokumen karena kemunculannya tidak unik atau tidak pernah digunakan di dalam masukkan pencarian. Untuk itu dilakukan proses penyaringan kata-kata tersebut. Penyaringan dilakukan dengan menyediakan sebuah daftar kata-kata yang tidak penting diindeks (*stopword list*). Tidak ada aturan pasti dalam menentukan *stopword* yang akan digunakan.

2.8 *Stemming*

Menurut Tala (2003) *stemming* merupakan suatu proses yang terdapat dalam sistem IR yang mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (*rootword*) dengan menggunakan aturan-aturan tertentu. Sebagai contoh, kata “melupakan”, “dilupakan” dapat dikatakan serupa atau satu kelompok dan dapat mewakili oleh satu kata umum “lupa”. Proses *stemming* pada teks Bahasa Indonesia berbeda dengan teks berbahasa Inggris. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks.

Sedangkan pada teks berbahasa Indonesia, selain sufiks, prefiks, dan konfiks juga dihilangkan. Menurut Nazief & Andriani (2007) Algoritma stemming untuk beberapa bahasa telah dikembangkan, seperti Algoritma Porter untuk teks berbahasa inggris, Algoritma Porter untuk teks berbahasa Indonesia, Algoritma Nazief & Adriani untuk teks berbahasa Indonesia.

2.9 Pembobotan TF-IDF

Setelah melalui tahap *preprocessing text*, data yang masih berupa *text* harus diubah ke dalam bentuk numerik. Menurut Simatupang & Utomo (2019) metode pembobotan TF-IDF disini dapat digunakan untuk mengubah data yang berbentuk teks menjadi numerik dengan menentukan hubungan masing-masing kata (*term*) pada suatu dokumen dengan cara memberikan bobot pada masing-masing *term* tersebut. Metode ini menggabungkan dua konsep untuk perhitungan bobot, yaitu *Term Frequency* (TF) merupakan frekuensi kemunculan kata (t) pada kalimat (d). *Document Frequency* (DF) adalah banyaknya kalimat dimana suatu kata (t) muncul.

Menurut Putra (2016) frekuensi kemunculan kata di dalam dokumen yang diberikan menunjukkan seberapa penting kata itu di dalam dokumen tersebut. Frekuensi dokumen yang mengandung kata tersebut menunjukkan seberapa umum kata tersebut. Bobot kata semakin besar jika sering muncul dalam suatu dokumen dan semakin kecil jika muncul dalam banyak dokumen. Pada TF-IDF digunakan rumus untuk menghitung bobot (W) masing-masing dokumen terhadap kata kunci dengan rumus yaitu:

$$W_{dt} = tf_{dt} * IDF_t \dots\dots\dots\dots\dots (1)$$

Dimana :

d = dokumen ke-d

t = kata ke-t dari kata kunci

W = bobot dokumen ke-d terhadap kata ke-t

tf = banyaknya kata yang dicari pada sebuah dokumen

IDF = *Inverse Document Frequency*

df = banyak dokumen yang mengandung kata yang dicari

Menurut Amin (2012) setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses *sorting*/pengurutan dimana semakin besar nilai W, semakin besar tingkat similaritas dokumen tersebut terhadap kata kunci, demikian sebaliknya. *Inverse Document Frequency* memperhatikan kemunculan *term* pada kumpulan dokumen. Pada metode ini, *term* yang dianggap bernilai/berharga adalah *term* yang jarang muncul pada koleksi/kumpulan dokumen. Persamaan IDF adalah sebagai berikut:

$$IDF_t = log\left(\frac{n}{df_t}\right) \dots\dots\dots\dots\dots (2)$$

df_t adalah banyak dokumen yang mengandung *term* t.

TF*IDF merupakan kombinasi metode TF dengan metode IDF. Sehingga persamaan TF*IDF adalah sebagai berikut:

$$TF * IDF_{dt} = TF_{dt} * IDF_t \dots\dots\dots\dots\dots (3)$$

Perhitungan bobot *input*an *relevance* merupakan bobot hasil perbandingan kemiripan (similaritas) antara *input*an yang dimasukkan oleh pengguna terhadap keseluruhan kalimat. Sedangkan bobot *similarity* kalimat, merupakan bobot hasil perbandingan kemiripan antar kalimat.

2.10 *Cosine Similarity*

Menurut Cios et al (2016) metode similaritas yang digunakan untuk menghitung similaritas dua buah dokumen. Metode yang dipergunakan adalah melakukan perhitungan ukuran kesamaan antara dua buah vektor dalam sebuah ruang dimensi yang didapat dari nilai cosinus sudut dari perkalian dua buah vektor yang dibandingkan karena cosinus dari 0 adalah 1 dan kurang dari 1 untuk nilai sudut yang lain, maka nilai similarity dari dua buah vektor dikatakan mirip ketika nilai dari *Cosine Similarity* adalah 1. Menurut Zizka et al (2019) *Cosine Similarity*

bisa hanya dihitung sebagai nilai rentang *Cosine Similarity* dari 0 hingga 1 sebagai sudut antara vektor dapat berada diantara 0 sampai 90 derajat, semakin mirip dokumennya maka semakin kecil sudutnya diantara *vector*. Berikut adalah rumus *Cosine Similarity.*

$$Similarity = cos(\theta) = \frac{A \bullet B}{\|A\| \, \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \dots\dots\dots\dots\dots (4)$$

Keterangan :

A = vektor A, yang akan dibandingkan kemiripannya

B = vektor B, yang akan dibandingkan kemiripannya

$\|A\|$ = panjang vektor A

$\|B\|$ = panjang vektor B

A_i = bobot *term* i dalam blok A_i

B_i = bobot *term* i dalam blok B_i

i = jumlah *term* dalam kalimat

n = jumlah vektor

Dimana A merupakan bobot setiap ciri pada *vector* A, dan B merupakan bobot setiap ciri pada B. Jika dikaitkan dengan *information retrieval* maka A adalah bobot setiap istilah pada dokumen A dan B merupakan bobot setiap istilah pada dokumen B.

2.11 Evaluasi

Menurut Mastur (2012) evaluasi digunakan untuk mengukur kinerja suatu sistem demi menghasilkan perbaikan pada proses pengambilan informasi. Ukuran umum yang digunakan untuk mengukur kualitas dari *text retrieval* adalah kombinasi *precision* dan *recall*. Metode yang umum digunakan adalah *recall*, *precision*, dan *f-measure*.

1. *Recall*

Recall adalah proporsi jumlah dokumen teks yang relevan terkenali diantara semua dokumen teks relevan yang ada pada koleksi. Rumus *recall* adalah

sebagai berikut:

$$Recall = \frac{jumlah\ dokumen\ relevan\ yang\ didapat\ sistem}{jumlah\ dokumen\ dalam\ database} \ldots\ldots\ldots\ldots\ldots (5)$$

2. *Precision*

Precision adalah proporsi jumlah dokumen teks yang relevan terkenali diantara semua dokumen teks yang terpilih oleh aplikasi. Rumus *precision* adalah sebagai berikut:

$$Precision = \frac{jumlah\ dokumen\ relevan\ yang\ didapat\ sistem}{jumlah\ dokumen\ yang\ didapat\ sistem} \ldots\ldots\ldots\ldots\ldots (6)$$

3. *F-Measure*

F-Measure adalah nilai yang mewakili seluruh kinerja aplikasi yang merupakan rata-rata dari nilai *precision* dan *recall*. Rumus *F-Measure* dapat dilihat pada persamaan berikut:

$$F - Measure = \frac{2\ \times Recall \times Precision}{Recall + Precision} \ldots\ldots\ldots\ldots\ldots (7)$$

2.12 Alat Bantu Perancangan Aplikasi

2.12.1 *Data Flow Diagram* (DFD)

Menurut Fatta (2007) *Data Flow Diagram* (DFD) merupakan diagram yang digunakan untuk menggambarkan proses-proses yang terjadi pada sistem yang akan dikembangkan. Dengan model ini, data-data yang terlibat pada masing-masing proses dapat diidentifikasi. Menurut Kristanto (2008) *Data Flow Diagram* merupakan suatu model logika data atau proses yang dibuat untuk menggambarkan darimana asal data dan kemana tujuan data yang keluar dari sistem, dimana data disimpan, proses apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan dan proses yang dikenakan pada data tersebut.

Berdasarkan uraian di atas dapat disimpulkan bahwa *data flow diagram*

adalah yang digunakan untuk menggambarkan proses-proses yang terjadi pada aplikasi yang akan dikembangkan dan dapat menghasilkan data tersebut dan interaksi antara data tersimpan dan proses yang dikenakan pada data tersebut.

2.12.2 *Flowchart*

Menurut Indrajani (2011) *flowchart* merupakan penggambaran secara grafik dari langkah-langkah dan urutan prosedur suatu program. Biasanya mempengaruhi penyelesaian masalah yang khusunya perlu dipelajari dan dievaluasi lebih lanjut. Menurut Supardi (2013) *flowchart* merupakan bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi.

2.12.3 *Entity Diagram Relationship* (ERD)

Menurut Sutanta (2011) *Entity Relationship Diagram* (ERD) adalah suatu bentuk diagram yang digunakan untuk menggambarkan suatu model data yang dikembangkan berdasarkan objek. *Entity Relationship Diagram* (ERD) tersusun atas tiga komponen yaitu entitas, atribut, dan kerelasian antar entitas. Entitas merupakan objek dasar yang terlibat dalam sistem, atribut berperan sebagai penjelas entitas, sedangkan kerelasian menunjukkan hubungan yang terjadi diantara dua entitas.

2.12.4 Kamus Data

Menurut Sutabri (2012) kamus data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. *Data dictionary* tidak menggunakan notasi grafik sebagaimana halnya *data flow diagram*. Kamus data berfungsi untuk membantu *user* agar dapat mengerti aplikasi secara rinci.

Menurut Mustakini (2009) kamus data atau *data dictionary* adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan kamus data, analis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. Kamus data dibuat pada tahap analisis sistem

dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem.

2.13 Metode Pengujian Aplikasi

2.13.1 *Black Box Testing*

Menurut Mustaqbal et al (2015) *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak, kumpulan kondisi *input* dan melakukan pengetesan pada fungsional program. Menurut Syaban (2015) *Black Box Testing* adalah pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak.

Menurut Khan (2011) *Black Box Testing* merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, *tester* dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada spesifikasi fungsional program. Berdasarkan definisi-definisi tersebut dapat disimpulkan bahwa *Black Box Testing* adalah metode pengujian yang berfokus pada persyaratan fungsional dari perangkat lunak.

2.13.2 *User Acceptance Testing* (UAT)

Menurut Perry (2006) *User Acceptance Testing* merupakan pengujian yang dilakukan oleh *end-user* dimana *user* tersebut adalah *staff*/karyawan perusahaan yang langsung berinteraksi dengan sistem dan dilakukan verifikasi apakah fungsi yang ada telah berjalan sesuai dengan kebutuhan/fungsinya. Menurut Lewis (2009) setelah dilakukan *system testing*, *acceptance test* menyatakan bahwa sistem *software* memenuhi persyaratan. *Acceptance test* merupakan pengujian yang dilakukan oleh pengguna yang menggunakan teknik pengujian *black box* untuk menguji sistem terhadap spesifikasinya. Pengguna akhir bertanggung jawab untuk memastikan semua fungsionalitas yang relevan telah diuji.

Dari definisi di atas, *User Acceptance Test* adalah pengujian yang dilakukan oleh pengguna dari aplikasi baru untuk memastikan fungsi-fungsi yang terdapat pada aplikasi tersebut telah berjalan dengan baik dan sesuai dengan

kebutuhan pengguna. Proses dalam UAT adalah pemeriksaan dan pengujian terhadap hasil pekerjaan.