Sistem Retrieval e-Arsip Tirta Asasta Menggunakan Algoritma Vector Space Model

by Dimasdrajad624@gmail.com 1

Submission date: 02-Jul-2024 07:59AM (UTC-0400)
Submission ID: 2411610318
File name: JURNAL_BARU.docx (493.08K)
Word count: 3297
Character count: 21795

JURNAL FASILKOM
Volume XX No. X | Bulan YYYY: xx-yy

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Sistem Retrieval E-Arsip Tirta Asata Menggunakan Algoritma Vector Space Model

Iqbal Hanan Junaidi[1], Sopingi[2], Sri Sumarlinda[3]
[1]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[2]Sistem Informasi, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[3]Sistem Informasi, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[1]202020345@mhs.udb.ac.id *, [2]sopingi@udb.ac.id, [3]sri_sumarlinda@udb.ac.id

Abstract

PT. Tirta Asasta Depok, facing challenges in document management. Currently, PT. Tirta Asasta Depok is still using manual archiving, which has barriers in the effectiveness and efficiency of time when searching for documents. This system implements an effective and efficient records management system at PT. Tirta Asasta Depok by utilizing text mining techniques, particularly the Vector Space Model algorithm. The problem faced by the company at this time is the storage and search of documents that are still manual, causing irregularity in the archives and hindering productivity. The research method used is to analyze document data in pdf format owned by PT. Tirta Asasta Depok. The Vector Space Model algorithm is applied to perform document similarity searches based on keywords entered by the user. Through term weighting and the ability to match partial queries with existing documents, this algorithm is expected to improve the effectiveness and efficiency of the company's records management. The results of this research are expected to produce a more organized, easily accessible, and quickly retrievable records management system when needed. The application of text mining technology, particularly the Vector Space Model algorithm, has been proven to automate the archiving process and improve the overall performance of the organization.

Keywords: Archive Management, Text Mining, Vector Space Model, Information Retrieval, Retrieval System

Abstrak

PT. Tirta Asasta Depok, menghadapi tantangan dalam pengelolaan dokumen. Saat ini, PT. Tirta Asasta Depok masih menggunakan pengarsipan secara manual, yang memiliki hambatan dalam efektivitas dan efisiensi waktu saat mencari dokumen. Sistem ini mengimplementasikan sistem manajemen arsip yang efektif dan efisien di PT. Tirta Asasta Depok dengan memanfaatkan teknik text mining, khususnya algoritma Vector Space Model. Permasalahan yang dihadapi perusahaan saat ini adalah penyimpanan dan pencarian dokumen yang masih manual, sehingga menyebabkan ketidakteraturan arsip dan menghambat produktivitas. Metode penelitian yang digunakan adalah dengan menganalisis data dokumen dalam format pdf yang dimiliki oleh PT. Tirta Asasta Depok. Algoritma Vector Space Model diterapkan untuk melakukan pencarian kemiripan dokumen berdasarkan kata kunci yang dimasukkan oleh pengguna. Melalui pembobotan kata (term) dan kemampuan mencocokkan sebagian query dengan dokumen yang ada, algoritma ini diharapkan dapat meningkatkan efektivitas dan efisiensi dalam manajemen arsip perusahaan. Hasil dari penelitian ini diharapkan dapat menghasilkan sistem manajemen arsip yang lebih teratur, mudah diakses, dan dapat ditemukan dengan cepat saat dibutuhkan. Penerapan teknologi text mining, khususnya algoritma Vector Space Model, terbukti dapat mengotomatisasi proses pengarsipan dan meningkatkan kinerja organisasi secara keseluruhan.

Kata kunci: Manajemen Arsip, *Text Mining*, *Vector Space Model*, *Information Retrieval*, Sistem Temu Kembali

.

1. Pendahuluan

Dalam era digital yang terus berkembang, banyak instansi, termasuk PT. Tirta Asasta Depok, menghadapi tantangan dalam pengelolaan dokumen. Saat ini, PT. Tirta Asasta Depok masih menggunakan pengarsipan secara manual, yang memiliki hambatan dalam efektivitas dan efisiensi waktu saat mencari dokumen. Seharusnya, dokumen-dokumen tersebut disimpan dengan rapi, mudah diakses, dan dapat ditemukan dengan mudah ketika dibutuhkan. Oleh karena itu, implementasi sistem pengarsipan surat yang efisien dan efektif menjadi sangat penting[1]. Sistem pengarsipan idealnya dilengkapi dengan fitur pencarian yang dapat memudahkan dalam menemukan dokumen dengan cepat. Seiring waktu berjalan, jumlah dokumen yang perlu diarsipkan akan bertambah. Oleh karena itu, pengaturan arsip harus dilakukan secara teratur. Dengan menggunakan teknologi, pengarsipan dapat dilakukan secara otomatis dan menghasilkan manajemen dokumen yang efisien. Penggunaan teknologi memudahkan akses informasi dan mempercepat proses pencarian dokumen, sehingga sistem pengarsipan dapat lebih efektif dan efisien[2].

Pengelolaan dokumen dan arsip merupakan aspek krusial bagi perusahaan atau organisasi. Salah satu tantangan utamanya adalah mengenai penyimpanan dan pencarian dokumen yang masih manual. Kondisi ini bisa mengakibatkan ketidakteraturan dalam penyimpanan arsip, yang akhirnya dapat menghambat produktivitas dan efisiensi perusahaan. Oleh karena

P-ISSN : 2089-3353
E-ISSN : 2808-9162

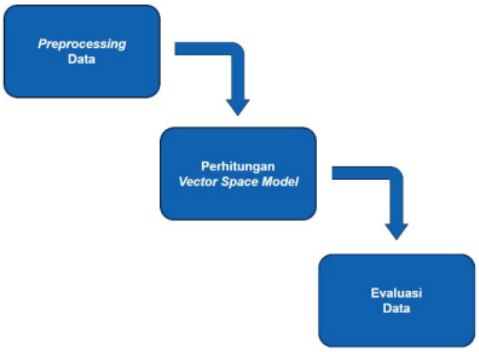
itu, dibutuhkan pengaturan yang baik dalam pengarsipan, dan teknologi dapat membantu mengotomatisasi proses pengarsipan untuk menciptakan manajemen dokumen yang efisien dan efektif. Pemanfaatan *text mining*, terutama dalam teknik *Information Retrieval* (IR), diyakini memiliki kemampuan untuk mendukung peningkatan manajemen arsip di perusahaan. Implementasi IR di perusahaan sangat krusial karena dapat terintegrasi dalam beragam fungsi, seperti menjelajah internet, perpustakaan digital, dan aplikasi pengarsipan[3]. Salah satu teknik dalam *Information Retrieval* yang bisa dimanfaatkan adalah *Vector Space Model*, yang unggul dalam kemampuan pencocokan *query* karena dapat mencocokkan bagian-bagian *query* dengan dokumen yang tersedia[4].

Ketidakteraturan dalam penyimpanan arsip dapat menghambat produktivitas dan efisiensi perusahaan. Saat harus mencari data, waktu yang dibutuhkan akan lebih lama dan hasilnya akan meluas dari topik pencarian yang sebenarnya dibutuhkan[5]. Oleh karena itu, penelitian ini dengan metode text mining bertujuan untuk meningkatkan manajemen arsip di PT. Tirta Asasta Depok.

Dengan menerapkan teknik Vector Space Model, penelitian ini dapat membantu dalam pengembangan sistem manajemen arsip yang lebih efektif dan efisien, yang pada gilirannya dapat meningkatkan performa keseluruhan organisasi. Dengan menggunakan SVM, diharapkan dapat mengukur tingkat kesamaan antara kata, frasa, dan dokumen melalui penggunaan bobot kata. Selain itu, model ini juga memungkinkan untuk melakukan pencocokan query dengan dokumen hanya sebagian. Sistem ini juga dapat disesuaikan dengan mudah melalui penyesuaian parameter, termasuk pengaturan bobot.

2. Metode Penelitian

Penelitian ini menggunakan data dokumen dari PT. Tirta Asasta Depok dalam bentuk pdf untuk melakukan pencarian kemiripan dengan sistem *retrieval*. Pengguna dapat memasukkan kata kunci ke dalam sistem, sehingga sistem dapat menemukan dokumen yang paling mirip. Proses ini dimulai dengan indeksasi dokumen berdasarkan seluruh isi dalam dokumen. Selanjutnya, dilakukan pengolahan teks yang mencakup tokenisasi dan penghapusan stopwords. Metode pembobotan yang diterapkan adalah TF-IDF dan nilai yang dihasilkan akan dimanfaatkan dalam perhitungan Model Ruang Vektor (VSM). Akhirnya, hasil akhirnya akan berupa tingkat kemiripan antara kata kunci dan dokumen yang disusun berdasarkan nilai tertinggi. Sistem aplikasi yang dimanfaatkan dalam penelitian ini dijelaskan dalam Gambar 1.



Gambar 1. Tahapan Metode Penelitian

2.1 *Preprocessing*

Preprocessing adalah tahapan yang sangat krusial karena dapat memengaruhi kualitas data. Pada tahap ini, dilakukan pemilihan fitur-fitur yang memiliki pengaruh signifikan terhadap prediksi yang akan dilakukan. [6]. Pada tahap ini dilakukan preprocessing pada saat text mining. Ketika sebuah dokumen atau teks mengandung banyak elemen non-standar seperti simbol dan angka, maka setiap teks akan dibagi menjadi unit-unit yang lebih kecil, biasanya kata-kata, dengan makna yang lebih detail. Langkah-langkah yang ditemui dalam proses pengolahan teks adalah sebagai berikut:

2.1.1 Tokenisasi

Tokenisasi merupakan proses memecah dokumen menjadi unit-unit terkecil yang disebut token. Dalam proses ini, teks dapat dibagi menjadi kata, simbol, frasa, dan elemen penting lainnya. Di samping itu, proses ini juga melibatkan penghapusan karakter khusus seperti simbol dan tanda baca, dan mengonversi semua huruf menjadi huruf kecil secara simultan[7].

2.1.2 Penghapusan *Stopwords*

Pembersihan stopword, khususnya tahap pengolahan data, dilanjutkan dengan pemilihan data yang dianggap dapat digunakan[8]. Kata-kata yang dianggap tidak relevan dihapus saat proses penghapusan kata sebelum dijadikan kata kunci dalam pencarian dokumen. Kata-kata yang dianggap tidak signifikan (*stopwords*) seperti kata-kata yang terdiri hanya 2 huruf kecuali kata hubung 'di', 'ke', 'dari', 'dan', 'atau', 'yang' akan disaring dan dimasukkan ke dalam daftar stoplists. Hal ini dilakukan untuk mengurangi jumlah kata dalam dataset guna meningkatkan kecepatan dan performa kerja sistem[7].

2.2 Perhitungan *Vector Space Model*

Setelah proses preprocessing selesai, dilakukan perhitungan kemiripan antara kata kunci dan dokumen menggunakan algoritma VSM dengan prosedur pembobotan terlebih dahulu. Metode pembobotan menggunakan *TF-IDF* untuk menghitung bobot dalam

P-ISSN : 2089-3353
E-ISSN : 2808-9162

proses pengambilan informasi. Setelah proses *preprocessing*, data akan diubah menjadi bentuk numerik menggunakan metode *TF-IDF*, di mana nilai *TF* berdasarkan frekuensi kata dalam dokumen.Jika suatu kata sering muncul maka nilai TFnya akan besar; jika suatu kata jarang muncul maka nilai TFnya akan kecil. Nilai IDF, di sisi lain, didasarkan pada frekuensi pencarian kata dalam kumpulan dokumen yang tersedia di database[7].

$$tf = tf_i \quad (1)$$

Perhitungan *Term Frequency* (*tf*)

Term frequency (*tf*) adalah suatu metode untuk mengukur seberapa sering sebuah term muncul dalam sebuah dokumen, sedangkan (tf_i) adalah banyaknya kemunculan *term* (t_i) dalam dokumen (d_j)

$$idf_i = log\frac{N}{dfi} \quad (2)$$

Perhitungan *Inverse Document Frequency* (*idf*)

idf_i adalah *inverse document frequency*, sedangkan N adalah jumlah dokumen dan df_i adalah banyaknya dokumen dalam koleksi Dimana *term* t_i muncul didalamnya.

$$W_{ij} = tf_{ij} \cdot log\frac{N}{dfi} \quad (3)$$

Perhitungan Term Frequency Inverse Document Frequency (*tfidf*)

W_{ij} adalah bobot dokumen yang dihitung, sedangkan tf_{ij} adalah banyaknya kemunculan *term* t_i pada dokumen d_j. N adalah jumlah dokumen dan df_i adalah banyaknya dokumen dalam koleksi dimana *term* t_i muncul didalamnnya.

$$|d_j| = \sqrt{\sum_{j=1}^{t}(Wij)^2} \quad (4)$$

Perhitungan Jarak Dokumen ($|d_j|$)

$|d_j|$ adalah jarak dokumen, sedangkan W_{ij} adalah bobot dokumen ke-i.

$$|q| = \sqrt{\sum_{j=1}^{t}(Wi,q)^2} \quad (5)$$

Perhitungan Jarak Query ($|q|$)

$|q|$ adalah jarak kueri, sedangkan $W_{i,q}$ adalah bobot kueri dokumen ke-i.

$$Sim(d,d_j) = \frac{dj \cdot q}{|dj| \cdot |q|} = \frac{\sqrt{\sum_{i=1}^{t} Wiq \cdot Wij}}{\sum_{i=1}^{t}(Wij)^2 \cdot \sum_{j=1}^{t}(Wi,q)^2} \quad (6)$$

Perhitungan Pengukuran Similaritas *Query Document*

W_{ij} adalah bobot *term* dalam dokumen. W_{iq} adalah bobot kueri, sedangkan Sim(q,dj) adalah similaritas antara kueri dan dokumen.

Sistem akan menghitung nilai kemiripan antara *query* pada dokumen arsip di PT. Tirta Asasta Depok. Nilai kemiripan tertinggi akan ditampilkan oleh sistem dalam bentuk niali similaritas. Nilai ini menunjukkan seberapa besar kesamaan atau kesesuaian antara kata kunci dengan dokumen arsip.

Dengan kata lain, sistem akan mengidentifikasi dokuken arsip yang paling dekat atau paling mirip dengan kata kunci yang telah dicari. Hasil identifikasi ini akan disajikan dalam bentuk nilai kemiripan, sehingga memudahkan pengguna untuk memahami seberapa relevan dokumen arsip tersebut dengan kata kunci yang telah dicari.

2.3 Evaluasi

Tujuan dari tahap ini adalah untuk menilai hasil kesamaan yang dihasilkan oleh algoritma, yang dicapai dengan membandingkan nilai kesamaan antar dokumen yang berbeda[9]. Dalam evaluasi ini, dilakukan evaluasi terhadap nilai-nilai seperti jarak antar dokumen, jarak antara query, dan nilai similaritas untuk mendapatkan pemahaman mengenai seberapa baik algoritma menghitung data penelitian.

3. Hasil dan Pembahasan

3.1 *Preprocessing*

Sistem akan melakukan pencarian dokumen yang diarsipkan berdasarkan kata kunci. Sebelum dilakukan pencarian, terlebih dahulu dilakukan proses pembersihan data teks yang akan digunakan. Adapun kata kunci yang akan diuji ialah "Pembinaan rohani pegawai dan permohonan dinas pendampingan umroh". Kata kunci tersebut kemudian diuji pada data uji. Penelitian ini menggunakan data eksperimen dari dua makalah yang ditunjukkan pada Gambar 2 dan 3.

Gambar 2. Data Penelitian 1

Gambar 3. Data Penelitian 2

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Setiap data uji sistem terlebih dahulu menjalani langkah prapemrosesan, termasuk pengkodean dan penghapusan kata-kata penghenti, kemudian proses pembobotan dilakukan menggunakan perhitungan *vector space model*.

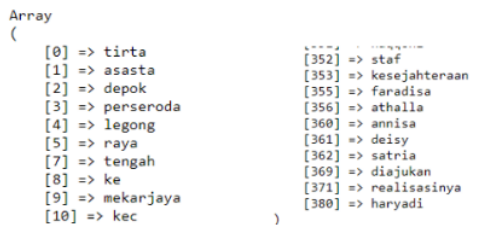
3.1.1 Tokenisasi

Output dari langkah ini adalah kalimat dari dokumen yang telah dibagi menjadi kata-kata, seperti yang terlihat pada Gambar 4.

Gambar 4. Hasil Tokenisasi

3.1.2 Penghapusan *Stopwords*

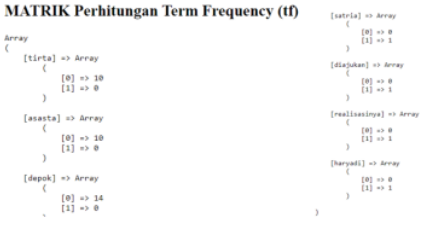
Pada tahap ini, sistem melakukan penghilangan kata-kata yang tidak memiliki makna atau tidak relevan, seperti yang terlihat pada ilustrasi dalam Gambar 5.



Gambar 5. Hasil Penghapusan *Stopwords*

3.2 Perhitungan *Vector Space Model*

Penelitian ini dilkakukan pencarian kemiripan dokumen dengan kata kunci yang telah dicari yaitu "Pembinaan rohani pegawai dan permohonan dinas pendampingan umroh".



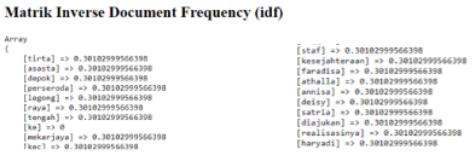
Gambar 6. Hasil Perhitungan TF

```
Function Terms Frequency
static function tf($terms, $dokumen){
  $jml_dokumen = count($dokumen);
  $matrik_tf = array();
  foreach ($terms as $item) {
    if (!empty($item)) {
      for ($i = 0; $i < $jml_dokumen; $i++) {
        $jml = substr_count(strtolower($dokumen[$i]['keyword']), $item);
        $matrik_tf[$item][$i] = $jml;
      }
    }
  }
  return $matrik_tf;
}
```

Fungsi tf tersebut menghitung term frequency (TF) untuk setiap istilah dalam daftar terms di setiap dokumen dalam koleksi. Hasil dari proses ini adalah sebuah matriks TF yang menunjukkan frekuensi kemunculan setiap istilah dalam setiap dokumen. Struktur matriks tersebut adalah sebagai berikut :

- matrik_tf[istilah][dokumen_index]: Frekuensi kemunculan istilah dalam dokumen yang diindeks oleh dokumen_index.

Fungsi ini memastikan bahwa setiap istilah dihitung secara case-insensitive dan hanya menghitung istilah yang tidak kosong.



Gambar 7. Hasil Perhitungan IDF

```
Function IDF
static function idf($matrik_tf, $dokumen){
  $N = count($dokumen);
  $matrik_df = array();
  $matrik_idf = array();
  foreach ($matrik_tf as $key => $item){
    $matrik_df[$key] = 0;
    foreach ($item as $value) {
      if ($value > 0) {
        $matrik_df[$key]++;
      }
    }
  }
  foreach ($matrik_df as $key => $value) {
    if ($value == 0) {
      $matrik_idf[$key] = 0;
    } else {
      $matrik_idf[$key] = log10($N / $value);
```

P-ISSN : 2089-3353
E-ISSN : 2808-9162

```
    }
  }
  return $matrik_idf;
}
```

Fungsi IDF menghitung nilai Inverse Document Frequency (IDF) untuk setiap istilah yang terdapat dalam kumpulan dokumen. IDF adalah metrik yang menunjukkan tingkat signifikansi suatu istilah dalam keseluruhan koleksi dokumen. Perhitungan IDF melibatkan jumlah dokumen dalam koleksi dan jumlah dokumen yang mengandung istilah tersebut. Dengan rumus yang digunakan $idf_i = log\frac{N}{dfi}$ dimana :

- N adalah jumlah total dokumen.
- DF adalah jumlah dokumen yang mengandung istilah tersebut.

Fungsi tersebut juga menangani kasus di mana nilai Document Frequency (DF) adalah 0 untuk menghindari pembagian dengan nol. Dalam kasus seperti itu, nilai Inverse Document Frequency (IDF) akan diatur menjadi 0. Hasil akhir dari proses ini adalah matriks IDF yang menunjukkan kepentingan relatif dari setiap istilah dalam keseluruhan koleksi dokumen.

```
[ashar] => Array
    (
        [0] => 0.30102999566398
        [1] => 0
    )

[berjamaah] => Array
    (
        [0] => 0.30102999566398
        [1] => 0
    )

[pembinaan] => Array
    (
        [0] => 1.8061799739839
        [1] => 0
    )

[rohani] => Array
    (
        [0] => 2.1072099696479
        [1] => 0
    )
```

Gambar 8. Hasil Perhitungan TFIDF

Function TFIDF

```
static function tfidf($matrik_tf, $matrik_idf) {
  $matrik_tfidf = array();
    foreach ($matrik_tf as $key => $item) {
      foreach ($item as $keyItem => $itemValue) {
        $matrik_tfidf[$key][$keyItem] = $itemValue * $matrik_idf[$key];
      }
    }
  return $matrik_tfidf;
}
```

Fungsi tfidf bertujuan untuk menghitung skor Term Frequency-Inverse Document Frequency (TF-IDF) untuk setiap istilah dalam setiap dokumen. Perhitungan nilai TF-IDF dilakukan dengan menggunakan matriks term frequency (TF) dan matriks inverse document frequency (IDF) yang telah diperhitungkan sebelumnya. TF-IDF merupakan ukuran yang sering dimanfaatkan dalam pemrosesan teks untuk mengevaluasi seberapa penting suatu istilah dalam dokumen tertentu, relatif terhadap keseluruhan koleksi dokumen. Dengan rumus yang digunakan $W_{ij} = tf_{ij} \cdot log\frac{N}{dfi}$ dimana :

- tf_{ij}, TF (Term Frequency): Mengukur seberapa sering suatu istilah muncul dalam sebuah dokumen.
- $log\frac{N}{dfi}$, IDF (Inverse Document Frequency): Mengukur seberapa penting istilah tersebut dalam seluruh koleksi dokumen.

TF-IDF diperoleh dengan mengalikan frekuensi term (TF) dengan frekuensi invers dokumen (IDF) untuk setiap istilah dalam setiap dokumen. Proses ini menghasilkan matriks TF-IDF yang menunjukkan bobot pentingnya setiap istilah dalam konteks masing-masing dokumen. Matriks TF-IDF ini dapat dimanfaatkan untuk berbagai keperluan pemrosesan teks, seperti pengukuran kesamaan dokumen, peringkatan dokumen, atau ekstraksi fitur yang relevan.

Perhitungan Jarak Dokumen (|dj|)

```
Array
(
    [0] => 9.322214162933
    [1] => 2.8078208759076
)
```

Gambar 9. Hasil Perhitungan Jarak Dokumen |dj|

Function |dj|

```
static function dj($matrik_tfidf, $dokumen){
  $matrik_w = array();
  $matrik_dj = array();
  $jml_dokumen = count($dokumen);
  for ($keyItem = 0; $keyItem < $jml_dokumen; $keyItem++) {
    $matrik_w[$keyItem] = 0;
  }
  foreach ($matrik_tfidf as $item) {
    foreach ($item as $keyItem => $itemValue){
      $kuadrat = $itemValue * $itemValue;
      $matrik_w[$keyItem] += $kuadrat;
    }
  }
  foreach ($matrik_w as $key => $item) {
    $matrik_dj[$key] = sqrt($item);
  }
  return $matrik_dj;
}
```

Fungsi dj tersebut menghitung panjang vektor (magnitude) dari nilai TF-IDF untuk setiap dokumen. Panjang vektor ini dihitung sebagai akar kuadrat dari

P-ISSN : 2089-3353
E-ISSN : 2808-9162

jumlah kuadrat nilai TF-IDF untuk semua istilah yang terdapat dalam dokumen tersebut. Nilai panjang vektor ini digunakan untuk normalisasi dokumen, yang merupakan langkah penting dalam berbagai algoritma pemrosesan teks dan pembelajaran mesin. Normalisasi dokumen berdasarkan panjang vektor TF-IDF memungkinkan perbandingan yang adil antara dokumen-dokumen, terlepas dari panjang teks masing-masing. Menggunakan rumus $|d_j| = \sqrt{\sum_{j=1}^{t}(Wij)^2}$.

Langkah-langkah perhitungannya sistem :

- Inisialisasi jumlah kuadrat nilai TF-IDF untuk setiap dokumen dengan 0.
- Iterasi melalui setiap istilah dalam matriks TF-IDF dan tambahkan kuadrat nilai TF-IDF ke jumlah kuadrat untuk setiap dokumen.
- Hitung akar kuadrat dari jumlah kuadrat nilai TF-IDF untuk mendapatkan panjang vektor untuk setiap dokumen.
- Kembalikan panjang vektor untuk setiap dokumen.

Hasil dari fungsi dj tersebut adalah sebuah array yang menunjukkan panjang vektor (magnitude) untuk setiap dokumen. Nilai-nilai ini dapat dimanfaatkan untuk proses normalisasi dokumen atau digunakan dalam perhitungan-perhitungan lebih lanjut dalam analisis teks. Normalisasi dokumen berdasarkan panjang vektor TF-IDF memungkinkan perbandingan yang adil antar dokumen, terlepas dari panjang teks masing-masing. Selanjutnya, array panjang vektor ini dapat diintegrasikan ke dalam berbagai algoritma pemrosesan teks dan pembelajaran mesin untuk menghasilkan analisis yang lebih akurat dan bermakna.

Perhitungan Jarak Query (|q|)

q = 0.79645050569775

Gambar 10. Hasil Perhitungan Jarak Query |q|

Function |q|

```
static function q($matrik_tfidf_query){

  $w_query = 0;

  foreach ($matrik_tfidf_query as $item) {

    $kuadrat = $item * $item;

    $w_query += $kuadrat;

  }

  $q = sqrt($w_query);

  return $q;
}
```

Fungsi q tersebut menghitung panjang vektor (magnitude) dari nilai TF-IDF untuk sebuah query. Panjang vektor ini dihitung sebagai akar kuadrat dari jumlah kuadrat nilai TF-IDF untuk semua istilah yang terdapat dalam query tersebut. Nilai panjang vektor ini digunakan untuk normalisasi query, yang merupakan langkah penting dalam berbagai algoritma pemrosesan teks dan pembelajaran mesin. Normalisasi query berdasarkan panjang vektor TF-IDF memungkinkan perbandingan yang adil antara query dan dokumen, terlepas dari panjang teks masing-masing. Hasil dari fungsi q adalah sebuah nilai yang menunjukkan panjang vektor (magnitude) untuk query, yang dapat dimanfaatkan dalam perhitungan-perhitungan lebih lanjut dalam analisis teks. Menggunakan rumus $|q| = \sqrt{\sum_{j=1}^{t}(Wi,q)^2}$.

Langkah-langkah perhitungan sistem :

- Inisialisasi jumlah kuadrat nilai TF-IDF untuk query dengan 0.
- Iterasi melalui setiap nilai TF-IDF dalam query dan tambahkan kuadrat nilai TF-IDF ke jumlah kuadrat.
- Hitung akar kuadrat dari jumlah kuadrat nilai TF-IDF untuk mendapatkan panjang vektor query.
- Tampilkan dan kembalikan panjang vektor query.

Hasil dari fungsi q adalah nilai yang menunjukkan panjang vektor (magnitude) dari nilai TF-IDF untuk query. Nilai panjang vektor ini dapat dimanfaatkan untuk proses normalisasi query atau digunakan dalam perhitungan-perhitungan lebih lanjut dalam analisis teks. Normalisasi query berdasarkan panjang vektor TF-IDF memungkinkan perbandingan yang adil antara query dan dokumen, terlepas dari panjang teks masing-masing. Selanjutnya, nilai panjang vektor ini dapat diintegrasikan ke dalam berbagai algoritma pemrosesan teks dan pembelajaran mesin untuk menghasilkan analisis yang lebih akurat dan bermakna.

Perhitungan dj.q / |dj|.|q|

```
Array
(
    [0] => 0.32953795369075
    [1] => 0.24313226954193
)
```

Gambar 11. Hasil Perhitungan Similaritas

Function Similaritas

```
static function sim($matrik_sum_dj_q, $matrik_djq){
```

P-ISSN : 2089-3353
E-ISSN : 2808-9162

```
    $matrik_sim = array();

    foreach ($matrik_sum_dj_q as $key => $item){

      $matrik_sim[$key]=$item/$matrik_djq[$key];

    }

    return $matrik_sim;

}
```

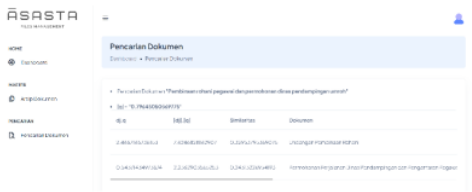
Fungsi sim menghitung nilai kesamaan (similarity) antara query dan setiap dokumen dalam koleksi. Perhitungan kesamaan ini dilakukan menggunakan rumus kosinus kesamaan. Rumus kosinus kesamaan mengukur sudut antara dua vektor, dalam konteks ini vektor TF-IDF dari query dan vektor TF-IDF dari masing-masing dokumen. Semakin kecil sudut antara dua vektor, semakin besar nilai kesamaan atau similaritasnya. Nilai kesamaan yang dihasilkan oleh fungsi sim menunjukkan tingkat kemiripan antara query dan setiap dokumen, yang dapat dimanfaatkan dalam berbagai aplikasi pemrosesan teks seperti perangkingan dokumen, temu kembali informasi, dan analisis topikalitas. Menggunakan rumus similarity $Sim(d,d_j)= \frac{dj \cdot q}{|dj| \cdot |q|} = \frac{\sqrt{\sum_{i=1}^{t} Wiq \cdot Wij}}{\sum_{i=1}^{t} (Wij)^2 \cdot \sum_{j=1}^{t} (Wi,q)^2}$, dimana :

- dj·q adalah hasil perkalian antara vektor query dan dokumen.
- |dj| adalah panjang vektor dokumen.
- |q| adalah panjang vektor query.

Hasil dari fungsi sim adalah sebuah array yang menunjukkan nilai kesamaan (similarity) antara query dan setiap dokumen dalam koleksi. Nilai-nilai kesamaan ini dihitung menggunakan rumus kosinus kesamaan, yang mengukur sudut antara vektor TF-IDF query dan vektor TF-IDF masing-masing dokumen. Semakin kecil sudut antara dua vektor, semakin besar nilai kesamaannya. Array nilai kesamaan yang dihasilkan dapat digunakan untuk menentukan dokumen yang paling relevan dengan query yang diberikan. Informasi ini kemudian dapat dimanfaatkan dalam berbagai aplikasi pemrosesan teks seperti meranking dokumen, pencarian informasi, dan analisis topik untuk menghasilkan hasil yang lebih tepat dan sesuai dengan kebutuhan pengguna.

3.3 Evaluasi

Sistem akan menghasilkan sebuah array yang menunjukkan panjang vektor (magnitude) dari nilai TF-IDF untuk setiap dokumen. Nilai-nilai panjang vektor ini dapat dimanfaatkan untuk proses normalisasi dokumen atau digunakan dalam perhitungan-perhitungan lebih lanjut dalam analisis teks. Selain itu, sistem juga akan menghasilkan nilai kesamaan (similarity) antara query dan setiap dokumen dalam koleksi. Nilai kesamaan ini dihitung menggunakan rumus kosinus kesamaan yang mengukur sudut antara vektor TF-IDF query dan vektor TF-IDF masing-masing dokumen. Array nilai kesamaan yang dihasilkan dapat digunakan untuk mengevaluasi relevansi dokumen dengan query yang diberikan. Informasi mengenai panjang vektor dan nilai kesamaan ini selanjutnya dapat diintegrasikan ke dalam berbagai aplikasi pemrosesan teks dan pembelajaran mesin untuk menghasilkan analisis similaritas yang lebih akurat.



Gambar 12. Implementasi Sistem Similaritas

4. Kesimpulan

Perusahaan PT. Tirta Asasta Depok masih menggunakan sistem pengarsipan manual yang tidak efisien dan tidak efektif. Oleh karena itu, disarankan untuk menerapkan sistem pengarsipan elektronik dengan menggunakan algoritma Vector Space Model guna meningkatkan pengelolaan dokumen perusahaan. Algoritma ini telah terbukti membantu pencarian dokumen dengan cepat dan akurat melalui pembobotan kata dan pencocokan query. Implementasi sistem pengarsipan elektronik berbasis algoritma ini juga dapat meningkatkan kinerja organisasi secara keseluruhan dengan mempercepat proses pencarian dan akses dokumen yang diperlukan.

Hasil penelitian menunjukan bahwa dengan menggunakan algoritma *Vector Space Model* dalam manajemen dokumen berdasarkan data yang telah diuji. Setelah pengujian diperoleh nilai similaritas di data pertama sebesar 0,32953795369075 dan data kedua diperoleh nilai similaritas sebesar 0,24313226954193. Hasil pencarian dapat diurutkan berdasarkan tingkat kesamaan yang dihasilkan oleh sistem, mulai dari nilai tertinggi hingga nilai terendah. Semakin tinggi nilai similaritas yang dihasilkan oleh sistem, semakin mirip kata kunci yang dicari dengan data dokumen arsip.

Ucapan Terimakasih

Penulis ucapkan terima kasih kepada PT Tirta Asasta Depok dan Universitas Duta Bangsa Surakarta yang telah membantu dalam kegiatan penelitian ini.

Daftar Rujukan

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Sistem Retrieval e-Arsip Tirta Asasta Menggunakan Algoritma Vector Space Model

ORIGINALITY REPORT

14%	13%	8%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.neliti.com Internet Source	2%
2	ejurnal.umri.ac.id Internet Source	2%
3	jik.htp.ac.id Internet Source	1%
4	socs.binus.ac.id Internet Source	1%
5	Serwin Gonzaga Rumagit, Debby Paseru, Thomas Ch. Suwanto. "Implementation of the Vector Space Model Algorithm for Document Searching in Material Test Monitoring Applications (Case Study: BPJN Jayapura Lab)", Jurnal Ilmiah Sains, 2024 Publication	1%
6	repository.teknokrat.ac.id Internet Source	1%
7	fr.scribd.com Internet Source	<1%

8	jurnal.fkmumi.ac.id Internet Source	<1 %
9	repository.trisakti.ac.id Internet Source	<1 %
10	e-journal.uajy.ac.id Internet Source	<1 %
11	jutif.if.unsoed.ac.id Internet Source	<1 %
12	www.scribd.com Internet Source	<1 %
13	core.ac.uk Internet Source	<1 %
14	jurnal-lp2m.umnaw.ac.id Internet Source	<1 %
15	medium.com Internet Source	<1 %
16	pt.scribd.com Internet Source	<1 %
17	repository.umj.ac.id Internet Source	<1 %
18	www.solidcircle.us Internet Source	<1 %
19	www.yesjogja.com Internet Source	<1 %

20	Ferdy Febriyanto. "Sistem Penilaian Otomatis Jawaban Esai Dengan Menggunakan Metode Vector Space Model Pada Beberapa Perkuliahan Di Stmik Indonesia Banjarmasin", Respati, 2019 Publication	<1 %
21	Karter D. Putung, Arie S.M. Lumenta, Agustinus Jacobus. "PENERAPAN SISTEM TEMU KEMBALI INFORMASI PADA KUMPULAN DOKUMEN SKRIPSI", Jurnal Teknik Informatika, 2016 Publication	<1 %
22	Monica Widiasri, Ellysa Tjandra, Lisa Maria Chandra. "Peningkatan Kinerja Pencarian Dokumen Tugas Akhir Menggunakan Porter Stemmer Bahasa Indonesia dan Fungsi Peringkat Okapi BM25", Teknika, 2017 Publication	<1 %
23	adoc.pub Internet Source	<1 %
24	de.slideshare.net Internet Source	<1 %
25	ejournal.akprind.ac.id Internet Source	<1 %
26	etheses.uin-malang.ac.id Internet Source	<1 %

27	repository.uin-suska.ac.id Internet Source	<1%
28	repository.usd.ac.id Internet Source	<1%
29	udb.ac.id Internet Source	<1%
30	www.eumed.net Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

Sistem Retrieval e-Arsip Tirta Asasta Menggunakan Algoritma Vector Space Model

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8