Laporan Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri
Aplikasi Dot Product pada Sistem Temu Balik Informasi
Semester 1 2020/2021

Anggota kelompok:

13519050 Faris Hasim Syauqi
13519061 Randy Zakya Suchrady
13519197 Muhammad Jafar Gundari

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

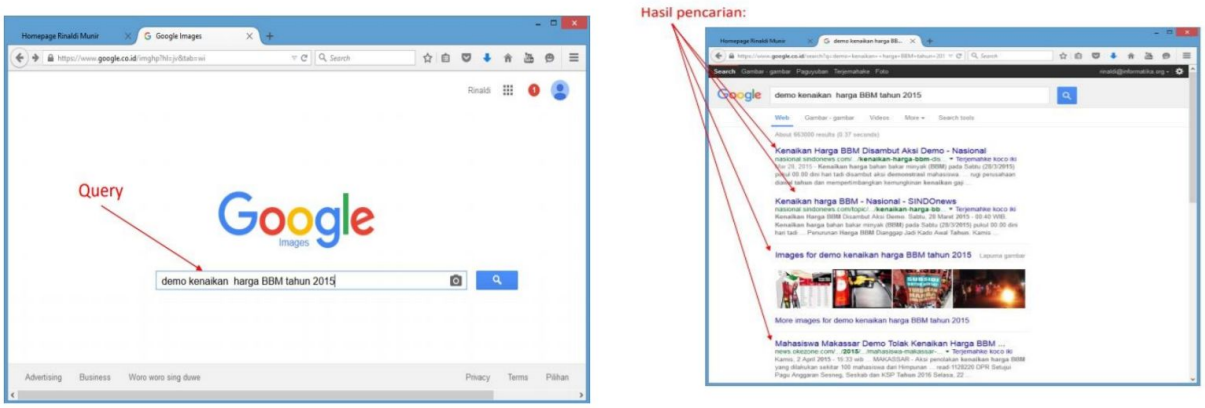
2020

BAB 1. Deskripsi Masalah

ABSTRAKSI

Hampir semua dari kita pernah menggunakan search engine, seperti google, bing dan yahoo! search. Setiap hari, bahkan untuk sesuatu yang sederhana kita menggunakan mesin pencarian Tapi, pernahkah kalian membayangkan bagaimana cara search engine tersebut mendapatkan semua dokumen kita berdasarkan apa yang ingin kita cari?

Sebagaimana yang telah diajarkan di dalam kuliah pada materi vektor di ruang Euclidean, temu-balik informasi (information retrieval) merupakan proses menemukan kembali (retrieval) informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Biasanya, sistem temu balik informasi ini digunakan untuk mencari informasi pada informasi yang tidak terstruktur, seperti laman web atau dokumen.



Gambar 1. Contoh penerapan Sistem Temu-Balik pada mesin pencarian
sumber: Aplikasi Dot Product pada Sistem Temu-balik Informasi by Rinaldi Munir

Ide utama dari sistem temu balik informasi adalah mengubah search query menjadi ruang vektor Setiap dokumen maupun query dinyatakan sebagai vektor w = (w1, w2,..., wn) di dalam Rn , dimana nilai wi dapat menyatakan jumlah kemunculan kata tersebut dalam dokumen (term frequency). Penentuan dokumen mana yang relevan dengan search query dipandang sebagai pengukuran kesamaan (similarity measure) antara query dengan dokumen. Semakin sama suatu vektor dokumen dengan vektor query, semakin relevan dokumen tersebut dengan query. Kesamaan tersebut dapat diukur dengan cosine similarity dengan rumus:

$$sim(\mathbf{Q}, \mathbf{D}) = \cos\theta = \frac{\mathbf{Q} \cdot \mathbf{D}}{\|\mathbf{Q}\|\|\mathbf{D}\|}$$

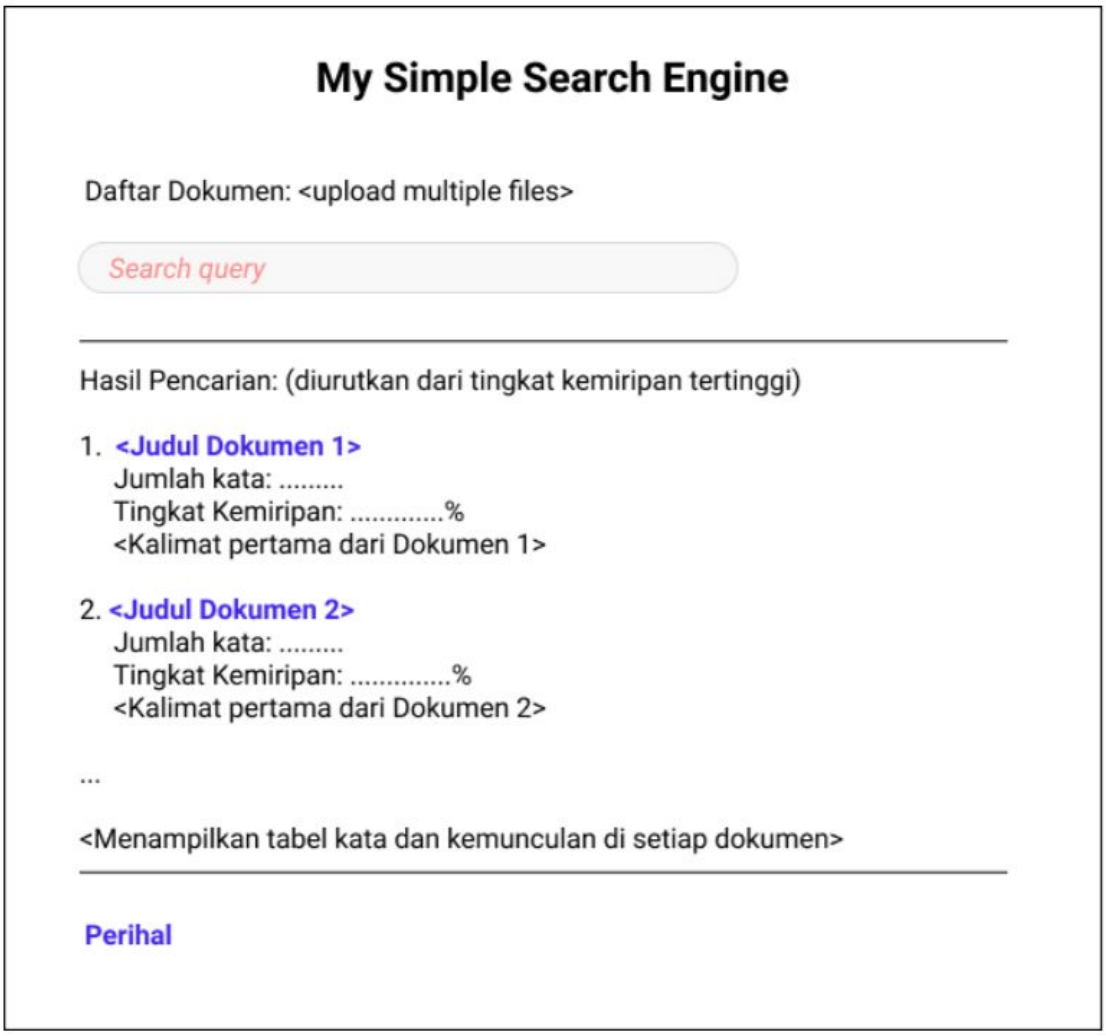
Pada kesempatan ini, kalian ditantang untuk membuat sebuah search engine sederhana dengan model ruang vektor dan memanfaatkan cosine similarity.

PENGGUNAAN PROGRAM

Berikut ini adalah input yang akan dimasukkan pengguna untuk eksekusi program.

1. **Search query**, berisi kumpulan kata yang akan digunakan untuk melakukan pencarian
2. **Kumpulan dokumen**, dilakukan dengan cara mengunggah multiple file ke dalam web browser.

Tampilan layout dari aplikasi web yang akan dibangun adalah sebagai berikut.



Gambar 2. Tampilan layout dari aplikasi web search engine yang dibangun.

Perihal: link ke halaman tentang program dan pembuatnya (Konsep singkat search engine yang dibuat, How to Use, About Us).

Catatan: Teks yang diberikan warna biru merupakan hyperlink yang akan mengalihkan halaman ke halaman yang ingin dilihat. Apabila menekan hyperlink , maka akan diarahkan pada sebuah halaman yang berisi full-text terkait dokumen 1 tersebut (seperti Search Engine).

Anda dapat menambahkan menu lainnya, gambar, logo, dan sebagainya. Tampilan Front End dari website dibuat semenarik mungkin selama mencakup seluruh informasi pada layout yang diberikan di atas.

Data uji berupa dokumen-dokumen yang akan diunggah ke dalam web browser. Format dan extension dokumen dibebaskan selama bisa dibaca oleh web browser (misalnya adalah dokumen dalam bentuk file txt atau file html). Minimal terdapat 15 dokumen berbeda.

Tabel term dan banyak kemunculan term dalam setiap dokumen akan ditampilkan pada web browser dengan layout sebagai berikut.

Term	Query	D1	D2	...	D3
Term1					
Term2					
...					
TermN					

Untuk menyederhanakan pembuatan search engine, terdapat hal-hal yang perlu diperhatikan dalam eksekusi program ini.

1. Silahkan lakukan stemming dan penghapusan stopwords pada setiap dokumen
2. Tidak perlu dibedakan antara huruf-huruf besar dan huruf-huruf kecil.
3. Stemming dan penghapusan stopword dilakukan saat penyusunan vektor, sehingga halaman yang berisi full-text terkait dokumen tetap seperti semula.
4. Penghapusan karakter-karakter yang tidak perlu untuk ditampilkan (jika menggunakan web scraping atau format dokumen berupa html)
5. Bahasa yang digunakan dalam dokumen adalah bahasa Inggris atau bahasa Indonesia (pilih salah satu)

Silahkan gunakan library sastrawi atau nltk untuk stemming kata dan penghapusan stopwords.

SPESIFIKASI TUGAS

Buatlah program mesin pencarian dengan sebuah website lokal sederhana. Spesifikasi program adalah sebagai berikut:

1. Program mampu menerima search query. Search query dapat berupa kata dasar maupun berimbuhan.
2. Dokumen yang akan menjadi kandidat dibebaskan formatnya dan disiapkan secara manual. Minimal terdapat 15 dokumen berbeda sebagai kandidat dokumen. Bonus: Gunakan web scraping untuk mengekstraksi dokumen dari website.

3. Hasil pencarian yang terurut berdasarkan similaritas tertinggi dari hasil teratas hingga hasil terbawah berupa judul dokumen dan kalimat pertama dari dokumen tersebut. Sertakan juga nilai similaritas tiap dokumen.
4. Program disarankan untuk melakukan pembersihan dokumen terlebih dahulu sebelum diproses dalam perhitungan cosine similarity. Pembersihan dokumen bisa meliputi hal-hal berikut ini. a. Stemming dan Penghapusan stopwords dari isi dokumen. b. Penghapusan karakter-karakter yang tidak perlu.
5. Program dibuat dalam sebuah website lokal sederhana. Dibebaskan untuk menggunakan framework pemrograman website apapun. Salah satu framework website yang bisa dimanfaatkan adalah Flask (Python), ReactJS, dan PHP.
6. Kalian dapat menambahkan fitur fungsional lain yang menunjang program yang anda buat (unsur kreativitas diperbolehkan/dianjurkan).
7. Program harus modular dan mengandung komentar yang jelas.
8. Dilarang menggunakan library cosine similarity yang sudah jadi.

BAB 2. Teori Singkat

Information Retrieval

Sistem temu-balik informasi atau biasa dikenal dengan *Information retrieval system* adalah suatu cara atau sistem untuk menemukan kembali (*retrieval*) informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis.

Ada berbagai macam model dalam sistem temu-balik informasi, salah-satunya yaitu dengan menggunakan model ruang vektor, dengan memanfaatkan berbagai teori dalam aljabar vektor.

Query dan setiap dokumen akan dinyatakan sebagai suatu vektor W = (w1, w2, w3, ... , wn) di dalam Rn. Dimana wi menyatakan bobot kata i pada dokumen tersebut, dalam hal ini adalah frekuensi kemunculan kata tersebut pada dokumen. Dokumen yang paling relevan dengan query akan ditentukan berdasarkan perhitungan kesamaan (*similarity measure*).

Cosine Similarity

Kesamaan (sim) antara dua vektor Q = (q1 , q2 , …, qn) dan D = (d1 , d2 , …, dn) diukur dengan rumus *cosine similarity* yang merupakan bagian dari rumus perkalian titik (dot product) dua buah vektor:

$$sim(\mathbf{Q}, \mathbf{D}) = \cos\theta = \frac{\mathbf{Q} \cdot \mathbf{D}}{\|\mathbf{Q}\|\|\mathbf{D}\|}$$

Dengan Q . D adalah perkalian titik (dot product) dua buah vektor:

$$\mathbf{Q} \cdot \mathbf{D} = q_1 d_1 + q_2 d_2 + \ldots + q_n d_n$$

.

BAB 3. Implementasi Program

Garis Besar Program

Program yang dibuat menggunakan framework Flask, dengan bahasa pemrograman Python, HTML, dan CSS. Program terdiri dari dua buah file program. File app.py adalah file *source code* program utama serta penggunaan framework-nya. File fungsi.py adalah file *source code* yang berisikan berbagai fungsi dan prosedur untuk pengolahan dokumen, data, dan perhitungan *cosine similarity*.

Langkah-langkah pemrosesan teks :

1. Isi dari file .txt serta query dijadikan sebagai array of string. Sehingga jika ada =3 files yang di-*upload* dan sebuah query, maka terdapat array berisi 4 buah string. Dengan urutan query, file1, file2, file3.
2. Setiap elemen pada array of string tersebut kemudian dilakukan proses penghapusan tanda baca dan juga penghapusan *stopwords* , menggunakan fungsi *removePunctuation* dan *removeStopwords.*Digunakan library nltk python.
3. Selanjutnya string akan dilakukan stemming. Stemming ini berfungsi untuk menyeragamkan setiap kata yang memiliki kata dasar sama, namun dengan melakukan ini berpotensi mengubah maksud dari kalimatnya. Proses stemming ini menggunakan library dari nltk *SnowballStemmer.* Pada tahap ini array of string akan menjadi *array of array of word.*
4. Kemudian dilakukan pendataan kata unik yang muncul baik pada query ataupun isi file. Dijadikan array of string (berbentuk kata) yang kita sebut word_list.
5. Selanjutnya dibuat array 2 dimensi (seperti tabel) yang merepresentasikan jumlah kemunculan dari setiap kata unik. Urutan indeks pada baris disesuaikan dengan urutan indeks pada word_list (poin 4). Selanjutnya array 2 dimensi ini disebut sebagai word_data.
6. Pada poin 5, baik query ataupun string dari file sudah dapat diperlakukan sebagai vektor.
7. Selanjutnya dicari similiaritynya dengan menggunakan cosine similiarity, dengan memanfaatkan word_data dari poin 5. diimplementasikan dengan cara berikut
   a. Hitung panjang ‘vektor’ dari elemen word_data. kolom pada word_data merepresentasikan vektor dari query,file1,file2,file3.
   b. Hitung dot product query dengan file (diimplementasikan pada fungsi dot_product)
   c. Gunakan rumus cosine similarity
   d. Masukan ke dalam *array of sim*
   e. Terdapat pula array of rank yang digunakan untuk informasi pengurutan nilai similarity dari setiap file
8. Terakhir dilakukan pengiriman data-data dari tahap sebelumnya ke bagian *front-end*.

BAB 4. Ekperimen

Pada bab eksperimen kali ini, kami akan mencocokan perhitungan similarity dari search engine kami dengan perhitungan secara manual. Kami menggunakan 3 sampel file txt untuk mempermudah perhitungan eksperimen, untuk percobaan menggunakan file txt dengan jumlah kata yang banyak maka kami akan tampilkan gambar hasil pencariannya.
Kami telah menyiapkan file dengan nama “1.txt” , “2.txt” , “3.txt” , berikut adalah konten dari tiap-tiap file tersebut.

Kami akan mencoba mengisi query dengan kata “blue cake” , berikut adalah tampilan tabel hasil pencarian.

Term	Query	1.txt	2.txt	3.txt
blue	1	2	1	0
cake	1	3	2	1

Similiarity Table

Kami dapati perhitungan kemunculan term sudah benar dan tepat, berikut adalah hasil perhitungan kami dan juga hasil perhitungan search engine kami:

Vektor Query : (i + j)
Vektor 1.txt : (2i + 3j)
Vektor 2.txt : (i + 2j)
Vektor 3.txt : (j)

Kemudian kami hitung nilai similaritynya menggunakan formula similarity di bawah ini.

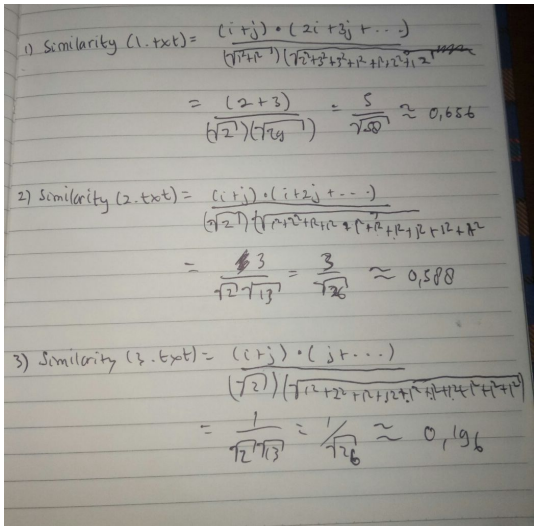
$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

***A** merupakan vektor Query dan **B** adalah vektor dokumen.
Dengan detail tabel term adalah sebagai berikut (tidak ditampilkan pada laman):

Term	Query	1.txt	2.txt	3.txt
blue	1	2	1	0
cake	1	3	2	1
want	0	3	0	0
someday	0	1	0	0
red	0	1	1	2
friend	0	2	0	0
need	0	1	0	1
color	0	0	1	1
wasnt	0	0	1	0
enough	0	0	1	0
yesterday	0	0	1	0
bought	0	0	1	0
100	0	0	1	0
million	0	0	1	0
new	0	0	0	1
beauti	0	0	0	1
birhtday	0	0	0	1
may	0	0	0	1
rainbow	0	0	0	1
colour	0	0	0	1

Similiarity Table

Maka perhitungan per dokumennya adalah:



Search result for : blue cake

1. Document Title : 1

- Total words : 22
- Similiarity : 0.66
- First sentence : i want blue cake, and he wants cake of blue.

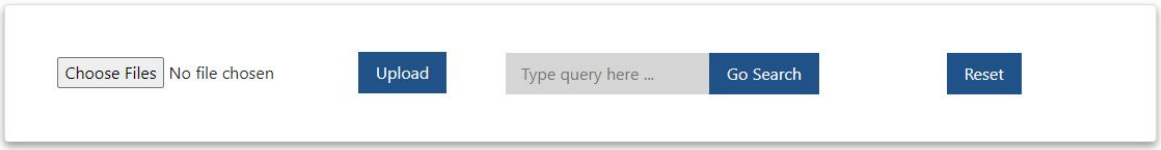
2. Document Title : 2

- Total words : 15
- Similiarity : 0.59
- First sentence : red colored cake wasn't enough for him.

3. Document Title : 3

- Total words : 21
- Similiarity : 0.2
- First sentence : A new cake that have beautiful color needs to be red.

Dari hasil perhitungan manual dan search engine didapat hasil yang cukup akurat, hanya berbeda di cara pembulatan. Berikut adalah eksperimen untuk file yang memiliki banyak kata di dalamnya.



Search result for : Donald Trump

1. Document Title : Trumps eldest children split on his path forward

- Total words : 1160
- Similiarity : 0.53
- First sentence : President Donald Trump has long sought advice from different perspectives ....

2. Document Title : What a Trump or a Biden win would mean for the Middle East

- Total words : 1661
- Similiarity : 0.28
- First sentence : An eerie calm hangs over the Middle East as the US presidential election ....

3. Document Title : How Chinas Xi Jinping blew a golden opportunity with US President Donald Trump

- Total words : 2397
- Similiarity : 0.22
- First sentence : At the first meeting between Donald Trump and Chinese leader Xi Jinping, ...

4. Document Title : Biden carries Arizona flipping a longtime Republican stronghold

- Total words : 1460
- Similiarity : 0.12
- First sentence : For just the second time in more than seven decades, a Democrat will carry ....

5. Document Title : In Central America a devastating storm and an uncertain future

*Untuk tabelnya cukup panjang

BAB 5. Kesimpulan, Saran, dan Refleksi

Kesimpulan

Pada tugas besar kali ini, kelompok kami dapat menyelesaikan pengerjaan program membuat website sederhana berupa *search engine* dengan memodelkan query dan *input file* sebagai vektor dan teori *cosine similiarity* untuk perhitungannya.

Saran

Pada tugas besar kali ini, kelompok sudah berhasil membuat algoritma perhitungan cosine similiarity dengan baik, diharapkan pengembang mampu meningkatkan fitur search engine ini, seperti memperluas ekstensi file yang diterima bukan hanya .txt saja, membuat web menjadi responsif untuk layar yang lebih kecil(seperti pada handphone).

Refleksi

Kami sebagai pengembang yang mengawali tugas besar ini dengan pengetahuan yang minim terhadap pengembangan website, mendapat pembelajaran terkait materi pengembangan website serta kerja sama dalam kelompok kami. Dari beban yang dilalui selama pengerjaan, seharusnya kami bisa mengerjakan tugas besar ini dengan waktu yang lebih cepat lagi untuk memperluas waktu sehingga dapat meningkatkan fitur-fitur yang ada pada website kami.

Referensi

1. http://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/algeo20-21.htm (Materi Kuliah Pak Rinaldi Munir)

2. https://pythonise.com/series/learning-flask/flask-uploading-files

3. https://www.tutorialspoint.com/flask/index.htm

4. https://www.geeksforgeeks.org/find-frequency-of-each-word-in-a-string-in-python/

5. https://medium.com/dev-genius/get-started-with-multiple-files-upload-using-flask-e8a2f5402e20

6. https://www.datacamp.com/community/tutorials/stemming-lemmatization-python