Penerapan LSA dan Query Suggestion untuk Pencarian Judul Artikel Menggunakan Framework FLASK

LSA and Query Suggestion for Article Searching with FLASK Framework

Komang Rinartha*[1], Luh Gede Surya Kartika[2]
[1]Institut Teknologi dan Bisnis STIKOM Bali, Denpasar, 80234, Indonesia
[2]Universitas Hindu Negeri I Gusti Bagus Sugriwa Denpasar, Denpasar, 80236, Indonesia
e-mail: *[1] **komangrinartha@gmail.com** , [2]suryakartika@uhnsugriwa.ac.id

Abstrak

Pencarian informasi melalui web sudah sangat berkembang seiring dengan dukungan perkembangan perangkat keras. Untuk mempermudah proses pencarian data, berbagai macam metode dikembangkan untuk pencarian informasi menggunakan data pencarian yang tersimpan pada browser ataupun menggunakan data yang terdapat di dalam database. Pengembangan query suggestion menggunakan Latent Semantic Analysis dapat dilakukan untuk mempermudah proses pencarian data tersebut. Penelitian ini dilakukan dengan menggunakan analisis sederhana metode Latent Semantic Analysis, Implementasi dalam bentuk program, Pengujian program dan analisis hasil query suggestion menggunakan metode tersebut. Data yang digunakan dalam penelitian ini adalah data publikasi yang telah dilaksanakan pada ICORIS 2019 sebagai data uji sebanyak 55 artikel. Sistem diimplementasikan berbasis web menggunakan FLASK framework dan bahasa pemrograman Python serta database MySQL. Adapun hasil dari penelitian ini adalah, semakin lengkap kata kunci yang dimasukkan, akan mendapatkan nilai similarity yang semakin tinggi untuk data target yang sesuai. Namun jumlah saran pencarian akan bertambah sesuai dengan data yang ada. Pada hasil penelitian, urutan kata kunci yang dicari tidak berpengaruh pada pemberian saran pencarian, karena data pencarian dan data didalam database dibandingkan berdasarkan kata-kata yang ada. Serta waktu proses yang didapatkan untuk pemberian saran pencarian adalah kurang dari 1 menit untuk 55 data judul yang digunakan.

Kata kunci— Query Suggestion, FLASK, Python, MySQL, Latent Semantic Analisys

Abstract

Searching information through the web has been very developed along with the support of hardware development. To simplify this process, various methods have been developed by using search data stored in the browser or using data in databases. The development of query suggestions using Latent Semantic Analysis can be done to simplify the search process. This research was conducted by analyzing the Latent Semantic Analysis, implementation, testing, and analysis of Latent semantic query suggestion result. The data used in this study are 55 articles that were pusblished in ICORIS 2019. The system was implemented on a web-based using the FLASK framework and the python programming language as well as the MySQL database. The results of this study are, the more complete the keywords entered, the higher similarity value for the appropriate target data. However, the number of suggestions will increase according to the existing data. In the results of the study, the order of the keywords had no effect on search suggestions, because the search data and the data in the database were compared based on the

existing words. Finally, the processing time obtained for giving suggestions is less than 1 minute for all articles.

Keywords— Query Suggestion, FLASK, Python, MySQL, Latent Semantic Analisys

1. PENDAHULUAN

Pencarian informasi melalui *website* sudah sangat berkembang seiring dengan dukungan perkembangan perangkat keras. Dalam perkembangannya, pencarian informasi dapat dilakukan dengan menggunakan data pencarian yang tersimpan pada *browser*. Selain itu, pencarian informasi juga dapat menggunakan data yang terdapat di dalam *database*. Google, Bing dan Yahoo! secara konsisten dilaporkan sebagai tiga mesin pencari web utama. Mereka memperkenalkan istilah *dynamic term/query suggestions* untuk mendukung pengguna dalam perumusan query, reformulasi atau perluasan query [1].

Untuk mempermudah proses pencarian data, berbagai macam metode dikembangkan. Salah satu metode yang dapat digunakan dalam proses pencarian data adalah Latent Semantic Analysis (LSA) yang dapat digunakan dalam *tag recommendation*[2], *movie recommendation*[3], *keyword extraction*[4] dan lain sebagainya. Sehingga pengembangan *query suggestion* menggunakan LSA dapat dilakukan untuk mempermudah proses pencarian data.

Query suggestion sudah banyak diterapkan pada website pencarian informasi untuk mempermudah proses pencarian informasi yang diinginkan. *Query suggestion* tidak lepas dari *form autocomplete* pada sebuah form *input* data pada *website*. *Autocomplete* merupakan pola yang pertama kali muncul dalam aplikasi *dekstop*, dimana pengguna memasukkan teks ke dalam kotak kemudian saran pengetikan akan muncul secara otomatis [5]. *Form autocomplete* merupakan fitur dalam sebuah form *input* untuk menampilkan data-data sesuai dengan data yang diketik oleh pengguna. Fitur *autocomplete* dengan *query suggestion* sering ditemui pada *search engine* dan juga form *input* dalam website.

Query suggestion yang kembangkan berupa aplikasi *website* yang mampu membantu pengguna dalam melakukan pencarian data. Website dibangun menggunakan teknik *responsive web design* yang mampu menyesuaikan tampilan sesuai dengan resolusi layar dan dengan bantuan Python sebagai *server-side scripting* serta MySQL sebagai penyimpanan datanya.

2. METODE PENELITIAN

Penelitian dilakukan dengan menggunakan metode analisis sederhana yaitu dengan menganalisis metode *Latent Semantic Analysis*, Implementasi, Pengujian program dan analisis *query suggestion* menggunakan metode tersebut. Sebelum melakukan analisis terhadap metode yang digunakan, dilakukan studi literatur untuk mengetahui penelitian terkait ataupun teknologi terkait dengan pengembangan query suggestion.

2.1 Literatur review

Dalam penelitian ini dilakukan penelusuran tentang penelitian lain yang pernah dilakukan dan masih relevan. Adapun penelitian itu tentang *fitur autocomplete*, *query suggestion*, *Latent Semantic Analysis*.

Fitur *autocomplete suggestion* merupakan layanan yang diterapkan pada perangkat lunak *web browser* dan mesin pencari yang memungkinkan pemberian saran pencarian pada pengguna melalui kata kunci yang dimasukkan oleh pengguna[6]. Kata kunci yang dimasukkan dapat berupa beberapa huruf ataupun beberapa kata. Fitur pada *autocomplete* Google dapat memberikan beberapa saran pencarian berdasarkan pada: letak geografis dan bahasa yang digunakan, kata kunci populer yang sering dicari, dan riwayat pencarian dari pengguna yang bersangkutan.

Autocomplete dari Google juga dapat memperbaiki kesalahan penulisan kata kunci saat pengguna memasukkannya menggunakan fitur *spelling correction* [7]. Selain di perangkat lunak *browser*, fitur ini juga dapat digunakan pada aplikasi *desktop*, *email-programs*, *search engine interface*, *source code editors*, *database query tools*, *word processor*, dan *command line interpreters* [5], maupun untuk aplikasi *game*[8].

Layanan *autocomplete* tersebut, sangat sering ditemukan terkait dengan layanan *query suggestion* untuk mempermudah proses pencarian data. *Query suggestion* merupakan layanan yang harus dimiliki oleh mesin pencari. Teknik tersebut dapat dibagi menjadi dua kategori berdasarkan data yang digunakan, yang pertama menggunakan data *log* dan yang kedua menggunakan data pencarian. *Query suggestion* berdasar pada data *log* dan data pencarian memiliki kelebihan dan kekurangan masing-masing sehingga mereka dapat dijalankan pada *query* yang berbeda[9]. Beberapa metode yang dapat digunakan dalam *query suggestion* antara lain, *MySQL pattern matching, Jaccard similarity* [10], *Levenshtein distance, MySQL Fulltext Index*[11] dan *Query expansion* [12].

Latent Semantic Analysis (LSA) merupakan teori dan metode yang digunakan untuk mengekstraksi dan mewakili makna penggunaan kata secara kontekstual dengan perhitungan statistik yang diterapkan pada kumpulan teks[13]. LSA merupakan metode yang mempunyai ciri khas hanya mementingkan kata-kata kunci yang terkandung dalam sebuah kalimat tanpa memperhatikan karakteristik linguistiknya. Pada LSA, kata-kata direpresentasikan dalam sebuah matriks *semantic* dan kemudian diolah secara matematis menggunakan teknik aljabar linier *Singular Value Decomposition* (SVD)[14]. SVD adalah sebuah metode untuk mengidentifikasi dan mengurutkan dimensi yang menunjukkan data dengan variasi yang paling banyak, sehingga hal ini memungkinkan untuk mecari pendekatan yang terbaik pada data asli menggunakan dimensi yang lebih kecil[14].

2. 2 Analisis Metode LSA

Metode LSA dianalisis dan diterapkan dalam bahasa pemrograman python dalam bentuk *command line*. Dalam proses analisis, digunakan sembilan kalimat yang diubah menjadi kumpulan kata yang menjadi pembentuk kalimat.

Tabel 1 Data yang digunakan dalam analisis

1)	"Qualitative Assessment of e-Government Implementation using PeGI Framework: Case Study Ministry of Marine Affairs and Fisheries the Republic of Indonesia",
2)	"Assessing Application Portfolios of IT Services through Maturity Levels of IT Governance",
3)	"Scientific Article Clustering Using String Similarity Concept",
4)	"Audit of Provincial Library Information System based on COBIT 4.1",
5)	"The Impact between the Use of Information Technology, User Ability on User Motivation and Employee Performance in the Koperasi Kuta Mimba",
6)	"The Mediating Role of Intention to Use E-Commerce Adoption in MSMEs",
7)	"Performance Comparison of Cat Swarm Optimization and Genetic Algorithm on Optimizing Functions",
8)	"Fire Incident Emergency Response Plan using Hybrid Fuzzy Dijkstra",
9)	"A New PHP Web Application Development Framework Based on MVC Architectural Pattern and Ajax Technology"

2. 3 Implementasi LSA

Setelah melakukan analisis terhadap metode LSA, kemudian diterapkan dalam sebuah aplikasi yang mudah untuk digunakan oleh pengguna dan mampu berinteraksi dengan *database* sesuai dengan data yang ada. Data yang digunakan dalam penelitian ini adalah data publikasi yang telah dilaksanakan pada ICORIS 2019 sebagai data uji. Data diambil secara langsung pada

website https://ieeexplore.ieee.org/xpl/conhome/8860912/proceeding sebanyak 55 artikel yang merupakan publikasi resmi ICORIS 2019.

Aplikasi dibangun menggunakan *FLASK framework* menggunakan Bahasa pemrograman Python. Dalam bidang *text mining*, Python memberikan kemudahan kepada *programmer* dalam membuat kode program menggunakan berbagai macam *library* yang ada. *Python Text mining package* berisi berbagai fungsi yang berguna untuk *text mining* di Python. Fokus utamanya pada *statistical text mining* dan membuatnya mudah untuk membuat istilah dalam dokumen dari kumpulan dokumen[15].

2. 4 Pengujian aplikasi

Aplikasi diuji menggunakan *black box testing* dan difokuskan pada kemampuan sistem untuk melakukan pencarian data yang relevan sesuai dengan kata kunci yang dimasukkan. Setelah melakukan pengujian *black box testing*, kemudian hasil pencarian menggunakan LSA dianalisis untuk mengetahui tingkat kesesuaian hasil pencarian.

3. HASIL DAN PEMBAHASAN

Penelitian ini menghasilkan sebuah aplikasi sederhana yang menggunakan algoritma LSA dalam melakukan pencarian data artikel berdasarkan kata kunci yang dimasukkan. Adapun hasil analisis metode yang didapatkan dijelaskan pada tahapan berikut:

3.1 Analisis Metode LSA

Dari kalimat-kalimat yang terdapat pada table 1 dipecah menjadi kumpulan kata, yang digambarkan menjadi sebuah array.

Tabel 2 Kumpulan kata

```
['ability' 'adoption' 'affairs' 'ajax' 'algorithm' 'application'
 'architectural' 'article' 'assessing' 'assessment' 'audit'
'based'
 'between' 'case' 'cat' 'clustering' 'cobit' 'commerce'
'comparison' 'concept' 'development' 'dijkstra' 'emergency'
'employee' 'fire' 'fisheries' 'framework' 'functions' 'fuzzy'
'genetic' 'governance' 'government' 'hybrid' 'impact'
'implementation' 'incident' 'indonesia' 'information' 'intention'
'it' 'koperasi' 'kuta' 'levels' 'library' 'marine' 'maturity'
'mediating' 'mimba' 'ministry' 'motivation' 'msmes' 'mvc' 'new'
'optimization' 'optimizing' 'pattern' 'pegi' 'performance' 'php'
'plan' 'portfolios' 'provincial' 'qualitative' 'republic'
'response' 'role' 'scientific' 'services' 'similarity' 'string'
'study' 'swarm' 'system' 'technology' 'through' 'use' 'user'
'using' 'web']
```

Dari kata-kata yang menjadi pembentuk masing-masing kalimat, kemudian dihilangkan kata yang termasuk dalam *stopword* dan dikumpulkan kembali menjadi kata yang bukan *stopword* dan diberikan lokasi dari kata tersebut. *Stopword* biasanya merupakan kata penghubung antar kalimat, dalam kasus ini *stopword* menggunakan kata-kata 'and','for','in','a','of','the','to', 'on'.

Tabel 3 Kumpulan kata dengan lokasi pada kalimat

```
{'qualitative': [0], 'assessment': [0], 'e-government': [0], 'implementation': [0], 'using': [0, 2, 7],
'pegi': [0], 'framework:': [0], 'case': [0], 'study': [0], 'ministry': [0], 'marine': [0], 'affairs': [0],
'fisheries': [0], 'republic': [0], 'indonesia': [0], 'assessing': [1], 'application': [1, 8], 'portfolios':
[1], 'it': [1, 1], 'services': [1], 'through': [1], 'maturity': [1], 'levels': [1], 'governance': [1],
```

'scientific': [2], 'article': [2], 'clustering': [2], 'string': [2], 'similarity': [2], 'concept': [2], 'audit': [3], 'provincial': [3], 'library': [3], 'information': [3, 4], 'system': [3], 'based': [3, 8], 'cobit': [3], '4.1': [3], 'impact': [4], 'between': [4], 'use': [4, 5], 'technology,': [4], 'user': [4, 4], 'ability': [4], 'motivation': [4], 'employee': [4], 'performance': [4, 6], 'koperasi': [4], 'kuta': [4], 'mimba': [4], 'mediating': [5], 'role': [5], 'intention': [5], 'e-commerce': [5], 'adoption': [5], 'msmes': [5], 'comparison': [6], 'cat': [6], 'swarm': [6], 'optimization': [6], 'genetic': [6], 'algorithm': [6], 'optimizing': [6], 'functions': [6], 'fire': [7], 'incident': [7], 'emergency': [7], 'response': [7], 'plan': [7], 'hybrid': [7], 'fuzzy': [7], 'dijkstra': [7], 'new': [8], 'php': [8], 'web': [8], 'development': [8], 'framework': [8], 'mvc': [8], 'architectural': [8], 'pattern': [8], 'ajax': [8], 'technology': [8]}

Kata “qualitative” terdapat pada kalimat pertama (indeks 0). Kata “using” terdapat pada kalimat pertama, ketiga dan ke delapan (indeks 0, 2 dan 7) begitu juga dengan kata lainnya. Kemudian dari kata-kata tersebut kemudian dibuatkan dalam bentuk matriks untuk kemunculan kata.

Tabel 4 Matriks kemunculan kata

```
[[0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 1]
 [0 0 0 0 1 0 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [1 0 0 0 0 0 0 0 0]
 [1 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [1 0 0 0 0 0 0 0 0]
 [0 0 0 1 1 0 0 0 0]
 [0 0 0 0 0 1 0 0 0]
```

```
  [0 2 0 0 0 0 0 0 0]
  [0 0 0 0 1 0 0 0 0]
  [0 0 0 0 1 0 0 0 0]
  [0 1 0 0 0 0 0 0 0]
  [0 0 0 1 0 0 0 0 0]
  [1 0 0 0 0 0 0 0 0]
  [0 1 0 0 0 0 0 0 0]
  [0 0 0 0 0 1 0 0 0]
  [0 0 0 0 1 0 0 0 0]
  [1 0 0 0 0 0 0 0 0]
  [0 0 0 0 1 0 0 0 0]
  [0 0 0 0 0 1 0 0 0]
  [0 0 0 0 0 0 0 0 1]
  [0 0 0 0 0 0 0 0 1]
  [0 0 0 0 0 0 1 0 0]
  [0 0 0 0 0 0 1 0 0]
  [0 0 0 0 0 0 0 0 1]
  [1 0 0 0 0 0 0 0 0]
  [0 0 0 0 1 0 1 0 0]
  [0 0 0 0 0 0 0 0 1]
  [0 0 0 0 0 0 0 1 0]
  [0 1 0 0 0 0 0 0 0]
  [0 0 0 1 0 0 0 0 0]
  [1 0 0 0 0 0 0 0 0]
  [1 0 0 0 0 0 0 0 0]
  [0 0 0 0 0 0 0 1 0]
  [0 0 0 0 0 1 0 0 0]
  [0 0 1 0 0 0 0 0 0]
  [0 1 0 0 0 0 0 0 0]
  [0 0 1 0 0 0 0 0 0]
  [0 0 1 0 0 0 0 0 0]
  [1 0 0 0 0 0 0 0 0]
  [0 0 0 0 0 0 1 0 0]
  [0 0 0 1 0 0 0 0 0]
  [0 0 0 0 1 0 0 0 1]
  [0 1 0 0 0 0 0 0 0]
  [0 0 0 0 1 1 0 0 0]
  [0 0 0 0 2 0 0 0 0]
  [1 0 1 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 1]]
```

Ketika pengguna melakukan pencarian menggunakan kata kunci “Koperasi Technology”, maka dari kumpulan data kata tersebut akan dihitung nilai kemiripan/*similarity* antara data yang dicari dengan data judul yang ada dengan membuat pencarian menjadi sebuah *vector*.

Tabel 5 Array kata yang dicari

```
[[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0]]
```

Kemudian dihitung nilai *similarity* antara data yang dicari dengan data yang ada di dalam matrix dan didapatkan nilai sebagai berikut:

Tabel 6 Nilai kemiripan

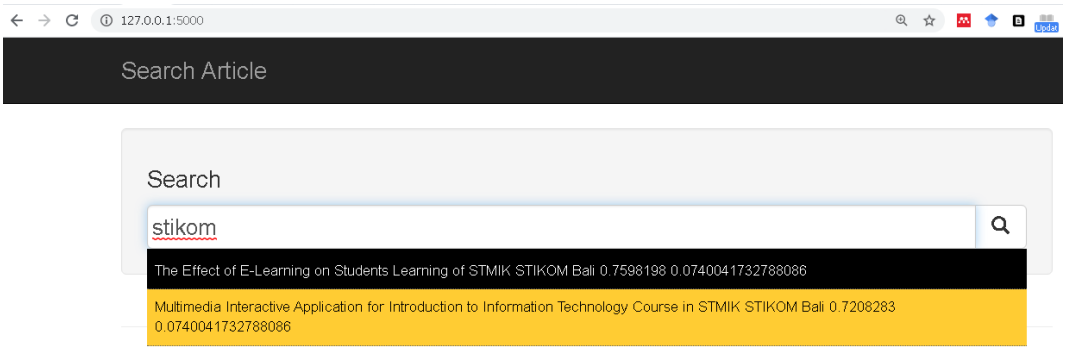
```
[-0.03438072]
```

```
[-0.04232895]
[0.00443622]
[-0.19040535]
[0.82489006]
[-0.11784144]
[-0.09165445]
[0.00332717]
[0.50794737]
```

Dari data tersebut, didapatkan nilai tertinggi adalah `0.82489006` yang merupakan data pada index ke 4 (data no. 5) dengan index pertama dimulai dari 0. Disimpulkan bahwa kata kunci "Koperasi Technology" akan sangat relevan sebagai saran pencarian adalah "The Impact between the Use of Information Technology, User Ability on User Motivation and Employee Performance in the Koperasi Kuta Mimba"

3.2 Implementasi LSA

Dari analisis yang dilakukan kemudian sistem diimplemetasikan menjadi sebuah website yang menggunakan *database* judul artikel pada ICORIS 2019 sebanyak 55 judul artikel.



Gambar 1 Tampilan pencarian data

Pada tampilan aplikasi, sistem mampu menampilkan hasil pencarian lengkap dengan nilai *similarity* dan waktu proses data dalam bentuk saran pencarian. Sehingga pengguna akan dibantu dengan menggunakan kata kunci yang minim tetapi menghasilkan saran pencarian yang tepat.

Pengujian pencarian data menggunakan target pencarian "The Effect of E-Learning on Students Learning of STMIK STIKOM Bali" pada aplikasi ditunjukkan pada tabel 7 dan tabel 8:

Tabel 7 Hasil waktu pencarian

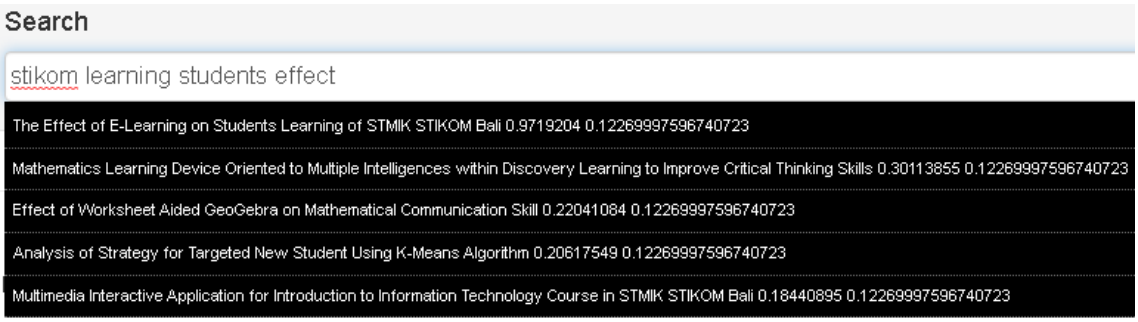
Kata kunci	***Similarity score* tertinggi**	**Waktu (detik)**
stikom	0.7598	0.121
stikom learning	0.9498	0.138
stikom learning students	0.9594	0.125
stikom learning students effect	0.9719	0.123

Tabel 8 Hasil pencarian

Kata kunci	Hasil Pencarian
stikom	Search stikom The Effect of E-Learning on Students Learning of STMIK STIKOM Bali 0.7598198 0.12135195732116699 Multimedia Interactive Application for Introduction to Information Technology Course in STMIK STIKOM Bali 0.7208283 0.12135195732116699
stikom learning	Search stikom learning The Effect of E-Learning on Students Learning of STMIK STIKOM Bali 0.9498134 0.13771414756774902 Mathematics Learning Device Oriented to Multiple Intelligences within Discovery Learning to Improve Critical Thinking Skills 0.49048153 0.13771414756774902 Multimedia Interactive Application for Introduction to Information Technology Course in STMIK STIKOM Bali 0.3003574 0.13771414756774902 Computer Application Utilization Based on Creative Problem Solving Model in Calculus Learning Implementation 0.28637958 0.13771414756774902 Distributed Training for Multilingual Combined Tokenizer using Deep Learning Model and Simple Communication Protocol 0.27418754 0.13771414756774902
stikom learning students	Search stikom learning students The Effect of E-Learning on Students Learning of STMIK STIKOM Bali 0.95945716 0.12499141693115234 Mathematics Learning Device Oriented to Multiple Intelligences within Discovery Learning to Improve Critical Thinking Skills 0.37159613 0.12499141693115234 Analysis of Strategy for Targeted New Student Using K-Means Algorithm 0.2544145 0.12499141693115234 Multimedia Interactive Application for Introduction to Information Technology Course in STMIK STIKOM Bali 0.22755522 0.12499141693115234 Computer Application Utilization Based on Creative Problem Solving Model in Calculus Learning Implementation 0.21696538 0.12499141693115234
stikom learning students effect	Search stikom learning students effect The Effect of E-Learning on Students Learning of STMIK STIKOM Bali 0.9719204 0.12269997596740723 Mathematics Learning Device Oriented to Multiple Intelligences within Discovery Learning to Improve Critical Thinking Skills 0.30113855 0.12269997596740723 Effect of Worksheet Aided GeoGebra on Mathematical Communication Skill 0.22041084 0.12269997596740723 Analysis of Strategy for Targeted New Student Using K-Means Algorithm 0.20617549 0.12269997596740723 Multimedia Interactive Application for Introduction to Information Technology Course in STMIK STIKOM Bali 0.18440895 0.12269997596740723

Pada tabel 7 dan tabel 8, terlihat bahwa semakin banyak dan semakin lengkap kata kunci yang dimasukkan akan mendapatkan *similarity score* yang semakin tinggi untuk data target yang sesuai. Namun jumlah saran pencarian akan bertambah sesuai dengan data yang ada, pada kasus ini hanya dibatasi lima data saran pencarian dengan *similarity score* tertinggi. Urutan kata kunci tidak berpengaruh pada pemberian saran pencarian, karena data pencarian dan data didalam *database* dibandingkan berdasarkan kata-kata yang ada dan bukan urutan kata.

Untuk waktu proses yang diperlukan adalah kurang dari satu menit untuk setiap proses pencarian. Ketika kata kunci yang dicari tidak terdapat pada judul-judul artikel, maka sistem tidak dapat menampilkan data yang dicari. Sistem yang dibangun juga belum memberikan hasil yang sesuai jika terdapat kesalahan pengetikan yang dilakukan dalam kata kunci pencarian.



Gambar 2 Tampilan hasil saran hasil pencarian

Pengujian *black box tesing* pada aplikasi dilakukan dengan memberikan data uji yang sesuai dengan kondisi yang diharapkan pada sistem. Adapun pengujian *blackbox testing* dilakukan sebagai berikut:

Tabel 9 Hasil pengujian black box

Data input	Hasil yang diharapkan	Hasil yang didapat	Status pengujian
Kata kunci ada dalam *database*	Sistem memberikan saran pencarian	Sistem memberikan saran pencarian	Sesuai
Kata kunci ada dalam *database*	Sistem memberikan nilai *similarity* dan waktu proses	Sistem memberikan nilai *similarity* dan waktu proses	Sesuai
Kata kunci lebih dari satu (lebih lengkap)	Nilai *similarity* semakin tinggi	Nilai *similarity* semakin tinggi	Sesuai
Kata kunci lengkap dan acak	Nilai *similarity* sama dengan kata kunci berurut	Nilai *similarity* sama dengan kata kunci berurut	Sesuai
Kata kunci tidak ada dalam *database*	Sistem tidak menampilkan saran pencarian	Sistem tidak menampilkan saran pencarian	Sesuai

4. KESIMPULAN

Berdasarkan hasil penelitian didapatkan beberapa kesimpulan antara lain, sistem yang dibangun menggunakan framework FLASK dan menggunakan metode *Latent Semantic Analysis* mampu memberikan saran pencarian sesuai dengan data yang ada didalam *database* serta mendapatkan hasil pencarian yang terurut berdasarkan *similarity score* dari data yang dicari dengan data yang terdapat didalam *database*. Semakin lengkap kata kunci yang akan dicari, maka *similarity score* yang dihasilkan akan semakin tinggi untuk data yang sesuai dan juga kemungkinan untuk menghasilkan saran pencarian lebih banyak karena kata yang digunakan sebagai acuan menjadi lebih banyak.

5. SARAN

Pada penelitian ini masih terbatas dengan pencarian data menggunakan data judul artikel sesuai yang tertera pada prosiding, sehingga penelitian dapat dilanjutkan dengan melakukan pencarian data berdasarkan pada keseluruhan tulisan yang terdapat dalam artikel. Selain itu penelitian juga dapat dilakukan dengan melakukan modifikasi metode LSA untuk menangani multi bahasa dalam sebuah prosiding dan juga dapat melakukan perbandingan antara metode LSA

dengan metode lainnya yang juga digunakan sebagai *query suggestion*. Penelitian lanjutan yang dapat dilakukan juga adalah penggunaan metode LSA dalam *natural language processing* dan juga dalam menangani *big data* ataupun *unstructured data* dalam berbagai bidang.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih yang sebesar besarnya kepada STIKOM Bali yang telah mengijinkan peneliti melakukan penelitian dan juga mengijinkan menggunakan data publikasi yang dilaksanakan oleh STIKOM Bali dan CORIS group dengan data publikasi yang diambil langsung melalui website resmi IEEE. Selain itu STIKOM Bali juga memberikan bantuan secara financial sehingga penelitian ini dapat terlaksana dengan baik.

DAFTAR PUSTAKA

[1] A. Shiri and L. Zvyagintseva, "Dynamic Query Suggestion in Web Search Engines: A Comparative Examination," *Proc. Annu. Conf. CAIS / Actes du congrès Annu. l'ACSI*, Jun. 2016, doi: 10.29173/cais868.

[2] I. Alagha and Y. Abu-Samra, "Tag recommendation for short arabic text by using latent semantic analysis of wikipedia," *Jordanian J. Comput. Inf. Technol.*, vol. 6, no. 2, pp. 165–181, Jun. 2020, doi: 10.5455/JJCIT.71-1575827721.

[3] K. Wegba, A. Lu, Y. Li, and W. Wang, "Interactive Movie Recommendation Through Latent Semantic Analysis and Storytelling," Jan. 2017, doi: 10.48550/arxiv.1701.00199.

[4] A. Makalesi and R. Article Fahrettin HORASAN, "Keyword Extraction for Search Engine Optimization Using Latent Semantic Analysis," *J. Polytech.*, vol. 24, no. 2, pp. 473–479, Jun. 2021, doi: 10.2339/POLITEKNIK.684377.

[5] Y. Primadani, "Simulasi Algoritma Levenshtein Distance Untuk Fitur Autocomplete Pada Aplikasi Katalog Perpustakaan," May 2014.

[6] D. Hawking and K. Griffiths, "An enterprise search paradigm based on extended query auto-completion. Do we still need search and navigation?," in *ACM International Conference Proceeding Series*, 2013, pp. 18–25, doi: 10.1145/2537734.2537743.

[7] L. Y. Banowosari, A. Darmawan, K. Kurniawan, and M. Mitchell, "Analisis Pada Fitur Autocomplete Suggestion Dan Semantik Pada Pencarian Di Mesin Pencari Google," in *Prosiding Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2014)*, 2014, vol. 8.

[8] M. Lee, T. B. Hashimoto, and P. Liang, "Learning Autocomplete Systems as a Communication Game," Nov. 2019, doi: 10.48550/arxiv.1911.06964.

[9] J. M. Yang, R. Cai, F. Jing, S. Wang, L. Zhang, and W. Y. Ma, "Search-based query suggestion," in *International Conference on Information and Knowledge Management, Proceedings*, 2008, pp. 1439–1440, doi: 10.1145/1458082.1458321.

[10] K. Rinartha and W. Suryasa, "Comparative study for better result on query suggestion of article searching with MySQL pattern matching and Jaccard similarity," 2017, doi: 10.1109/CITSM.2017.8089237.

[11] K. Rinartha, W. Suryasa, and L. G. S. Kartika, "Comparative Analysis of String Similarity on Dynamic Query Suggestions," 2018, doi: 10.1109/EECCIS.2018.8692996.

[12] S. Plansangket and J. Q. Gan, "A query suggestion method combining TF-IDF and Jaccard Coefficient for interactive web search," *Artif. Intell. Res.*, vol. 4, no. 2, Aug. 2015, doi: 10.5430/air.v4n2p119.

[13] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse Process.*, vol. 25, no. 2–3, pp. 259–284, Jan. 1998, doi: 10.1080/01638539809545028.

[14] R. B. Aji *et al.*, "Automatic Essay Grading System Menggunakan Metode Latent Semantic Analysis," *J. Inov. dan Kewirausahaan*, pp. 17–18, 2011.

[15] M. Anil Zende, M. Bhaskar Tuplondhe, S. Baban Walunj, and S. Vasudev Parulekar, "Text Mining Using Python," *Int. J. Curr. Eng. Sci. Res.*, no. 3, p. 54, 2016.