PENERAPAN ALGORITMA COSINE SIMILARITY PADA APLIKASI BANK SOAL

Daniel Nugraha[1]
[1]Fakultas Teknologi Informasi Universitas Widya Dharma, Pontianak
e-mail: [1] d1980n@gmail.com

Abstrak

Permasalahan dalam duplikasi data soal yang dimasukan pengguna dalam aplikasi bank soal merupakan salah satu permasalahan pengukuran kemiripan *text*. Setiap *text* soal yang dimasukan akan di ukur kemiripannya dengan *text* soal yang lain dalam *database*. Dalam penulisan ini, penulis menggunakan jenis penelitian terapan *Research and Development* (R&D). Rancangan penelitian menggunakan teknik analisis sistem, teknik perancangan sistem menggunakan *Unified Modeling* Language (UML) dan teknik perancangan aplikasi menggunakan bahasa pemrograman HTML, CSS, javascript, dan PHP. Pengembangan aplikasi menggunakan *web server* XAMPP dan MySQL sebagai tempat penyimpanan *database*.
Berdasarkan penelitian yang dilakukan, dihasilkan sebuah aplikasi bank soal dengan menerapkan algoritma *Cosine Similarity* untuk menampilkan selisih sudut antara text soal yang terakhir dimasukan dengan text soal yang lain dalam database. Sudut kemiripan *text* ditujukan jika selisih sudut sama dengan 0 maka kedua *text* tersebut memiliki kelompok kata yang sama dan jika sudut sama dengan 90 maka kelompok kata yang berbeda. Semakin besar selisih sudut maka akan semakin besar perbedaan *text* tersebut. Jika sudut semakin kecil, maka nilai Cosine akan semakin besar demikian juga sebaliknya. Kesimpulannya bahwa penggunaan aplikasi dapat memudahkan melihat perbandingan kemiripan antar data soal yang satu dengan soal yang lainnya dengan melihat nilai cosine yang ada.
Kata Kunci: Aplikasi, Cosine, Algoritma

Abstract

The Problematic of duplication in data question which was entered by user in question bank's application is one of general problem in text comparison. Every questions text that had been entered will compare with other text question in database. In this paper, the author is using research methods R&D (Research and Development). Research design techniques using Unified Modeling Language (UML) and application design techniques using the programming language HTML, CSS, javascript, and PHP. Application development using the XAMPP web server and MySQL as database storage.
Based on the research conducted, produce a question bank application system which implement Cosine similarity algorithm to show the difference in angle between last question text that was last entered in database with other question text in database. The similarity text angle is appearing 0 degrees if both texts are qual or had a same category text. If text angle is appearing 90 degrees, then both texts are not equal. The higher differentiation angle is higher differentiation text. The lower angle will produce higher cosine value. From this paper the conclusion, application will help to show the comparison between each data question with cosine value.
Keywords: Aplication, Cosine, Algorithms

1. PENDAHULUAN

Permasalahan terjadinya tindak plagiat dengan meng-*copy text* tanpa memasukan referensi yang ada pada era digital ini memang sudah menjadi permasalahan umum. *Text matching* atau yang biasa disebut dengan persamaan *text* merupakan salah satu cara untuk mengetahui apakah kalimat yang satu sama dengan kalimat yang lain. Metode untuk memecahkan masalah kesamaan *text* yang digunakan dan diteliti atara lain dapat di kelompokan menjadi empat pendekatan: *String-based, Corpus-base, Knowledge-base,* dan *Hybrid text similarities*[1]. Penelitian diawali dengan permasalahan adanya duplikasi soal yang dimasukan ternyata banyak yang sama. Kesamaan soal ini menyebabkan data yang disimpan akan bertambah banyak dalam database. Penelitian ini berfokus pada pendekatan *string-based* dengan metode *Cosine Similarity* [2].

Algoritma diterapkan pada aplikasi bank soal pada sistem yang berjalan menggunakan bahasa pemrograman HTML, CSS, dan JavaScript serta PHP sebagai *back end* program dan editor menggunakan

notepad++. Perancangan *database* dengan menggunakan MySql dalam web server XAMPP dan menggunakan Bootstrap sebagai *framework*. Teknik analisis yang digunakan untuk mengembangkan aplikasi adalah dengan menggunakan teknik berorientasi objek dengan bahasa pemodelan *Unified Modeling Languange* (UML) yang digunakan untuk menggambarkan proses kerja sistem yang ada.

Isi makalah ini dibagi menjadi beberapa bagian antara lain: bab 2 menjelaskan kajian teori yang menjelaskan mengenai algoritma yang digunakan, bab 3 implementasi dan pembahasan dari pemecahan permasalahan, bab 4 menampilkan hasil eksperimental dan bab 5 kesimpulan dan saran.

2. Kajian Teori

2.1 Text Similarity Algorithms

String-based similarity merupakan algoritma yang cukup popular. Perbandingan mengoperasikan string sekuen dan komposisi karakter. Metode *string-based* dibagi menjadi dua yaitu *character-based* dan *term-based*. *Character-based* menggunakan algoritma Smith Waterman, N-gram, Damerau-Lavenshtein, Jaro-Winkler, *Longgest Common Substring* (LCS), dan lain sebagainya sedangkan *term-based* menggunakan algoritma *block distance*, joccard *similarity*, *matching fefficient*, dan *overlap coefficient* [1]. Dalam pendekatan *mining,* algoritma yang digunakan untuk mencari informasi dalam text menggunakan algoritma seperti : *Information Retrieval* (IR), *text clasification, information extraction* (IE), *document clustering, sentiment analysus, machine translation, text summarization,* dan *natural language processing* (NLP)[4].

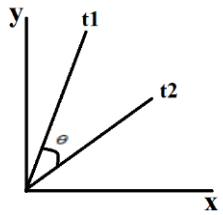
Kalimat yang memiliki kesamaan semantik dalam paragraf ataupun dalam dokumen dapat dihitung kesamaannya dengan menggunakan *word* vector *space* model (VSM) merupakan sebuah model ruang vektor yang diimplementasikan untuk semuah kalimat. Menurut Novotny, antar kalimat dapat di ukur dengan menggunakan pendekatan *Soft Cosine Measure* (SCM) yang memiliki kompleksitas waktu yang rendah[5].

2.2 *Cosine Similarity*

Cosine Similarity mengukur kesamaan dari vektor dalam dimensi yang sama [1]. *Cosine* merupakan cosinus sudut teta seperti pada persamaan berikut ini:

$$cos.\sin = \cos\theta$$

Sehingga dalam gambar 1 dijelaskan bahwa θ merupakan sudat diantara t1 dan t1.



Gambar 1. Contoh vektor 2 dimensi

Dari gambar 1 di atas, apabila sudut θ = 0^0 maka hasil cos 0 akan sama dengan 1. Demikian juga dengan cos 90 akan sama dengan 0. Vektor t1 dan vektor t2 jika memiliki sudut 0^0 maka $\overrightarrow{t1} = \overrightarrow{t2}$. Rumus untuk menentukan cos θ adalah sebagai berikut:

$$\cos\theta = \frac{\overrightarrow{t1} \,.\, \overrightarrow{t2}}{\left|\left|\overrightarrow{t1}\right|\right| \, ||\overrightarrow{t2}||}$$

$\overrightarrow{t1} \,.\, \overrightarrow{t2}$ merupakan perkalian sekalar antara vektor t1 dan vaktor t2. Jika vektor $\vec{a} = ax\hat{\imath} + ay\hat{\jmath}$ dan vektor $\vec{b} = bx\hat{\imath} + by\hat{\jmath}$, maka perkalian sekalar

$$\vec{a} \,.\, \vec{b} = ax\ bx + ay\ by$$

$\left|\left|\overrightarrow{t1}\right|\right|$ merupakan panjang vektor $\overrightarrow{t1}$. Rumusan menghitung panjang vektor adalah sebagai berikut:

$$||\vec{a}|| = \sqrt{ax^2 + ay^2}$$

2.3 Algoritma

Berikut ini merupakan algoritma *cosine* berdasarkan rumusan di atas.

```
function get_cosine(){
    return skalar(t1,t2) / (panjang(t1) * panjang (t2));
}
function skalar(t1, t2){
    hasil <- 0;
    for(i = 0; i < t1.lenght; i++){
        hasil <- hasil + (t1[i] * t2[i]);
    }
    return hasil;
}
function panjang(t){
    hasil <- 0;
    for (i = 0; i < t.length; i++){
        hasil <- hasil + (t[i] * t[i]);
    }
    hasil <- SQRT(hasil);
}
```

Gambar 2. Algoritma *Cosine*

Gambar 2 merupakan algoritma *cosine* yang didapat dari formula diatas. *Cosine* didapat dari hasil

bagi skalar (t1 dan t2) dengan hasil kali panjang t1 dan panjang t2. Skalar ke dua vektor merupakan hasil kali larik yang menyimpan nilai x dan y. panjang vektor t didapat dari akar kuadrat dari perkalian kuadrat vektor.

3. IMPLEMENTASI DAN PEMBAHASAN

3.1 *Text* dalam Vektor *Space*

Jika diberikan dua buah pertanyaan (soal) yang berupa *text* di lambangkan sebagai berikut: T = {t1, t2} dimana T ∈ D, D merupakan n-dimensi. Sehingga frekuensi vektor yang dibentuk dari T_n dalam D_n menjadi:

$$\overrightarrow{vD} = (f(D, t1), \dots, f(D, tn))$$

Sebagai contoh *text* sebagai berikut: jika t1 = "matahari terbit disebelah timur" dan t2 = "matahari terbenam disebelah barat atau disebelah timur" sehingga didapat T = {matahari, terbit, terbenam, disebelah, timur, barat, atau}. Sehingga jika vektor $\overrightarrow{t1} = (1,1,0,1,1,0,0)$ dan $\overrightarrow{t2} = (1,0,1,2,1,1,1)$, maka frekuensi vektor dapat digambarkan dengan tabel berikku ini:

Tabel 1. Frekuensi Vektor

	matahari	**terbit**	**terbenam**	**disebelah**	**Timur**	**barat**	**atau**
T1	1	1	0	1	1	0	0
T2	1	0	1	2	1	1	1

Dari hasil tabel 1 di atas, masukan rumus *cosine* dalam n-dimensi sebagai berikut:

$$\cos(A,B) = \frac{\sum_{i=1}^{n} A_i B_i}{\left|\left|\sqrt{\sum_{i=1}^{n} {A_i}^2}\right|\right| . \left|\left|\sqrt{\sum_{i=1}^{n} {B_i}^2}\right|\right|}$$

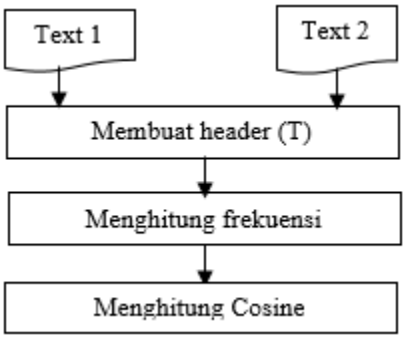
Dari perhitungan diatas didapat:

$$\cos(A,B) = \frac{1x1 + 1x0 + 0x1 + 1x2 + 1x1 + 0x1 + 0x1}{\sqrt{1^2+1^2+1^2+1^2}\ x\ \sqrt{1^2+1^2+2^2+1^2+1^2+1^2}}$$

$$\text{cosine} = \frac{4}{\sqrt{4}\ x\ \sqrt{9}} = \frac{4}{2\ x\ 3} = \frac{4}{6} = 0.667$$

3.2 Algoritma *Cosine* Terapan

Perbandingan *text* yang yang dinilai hasilnya adalah *text* pertanyaan soal dalam *database* yang terakhir di masukan dengan text pertanyaan soal yang lainnya. Algoritma *Cosine Block* diagram dalam aplikasi dapat diliat seperti pada Gambar 3 berikut ini:



Gambar 3. Algoritma *Cosine Block* Diagram

Langkah yang dilakukan sebelum menjalankan algoritma *cosine* adalah memasukan data terlebih dahulu minimal 2 pertanyaan soal sehingga bisa di bandingkan antara kedua *text*. Jika *text* yang dimasukan lebih dari 2, maka aplikasi akan mengulang dengan membandingkan dari *text* 1 sampai dengan *text* n-1. Hasil hitung *cosine* akan ditampilkan nilainya.

Gambar 3 menjelaskan secara sistematik bagaimana algoritma *cosine* diterapkan dalam aplikasi bank soal. Awal algoritma *cosine* dijalankan adalah dengan memasukan dua buah *text* yaitu *text* 1 dan *text* 2. Dari kedua buah *text* tersebut dapat dibentuk *header* dengan mengambil semua kata dalam *text* 1 dan *text* 2. Proses selanjutnya adalah dengan menghitung frekuensi untuk tiap-tiap kata yang terdapat dalam *text* 1 demikian juga pada *text* 2. Hasil frekuensi tersebut dapat dihitung perkalian skalar dan panjang *text*. *Cosine* di hitung dengan membagi skalar dengan panjang *text*.

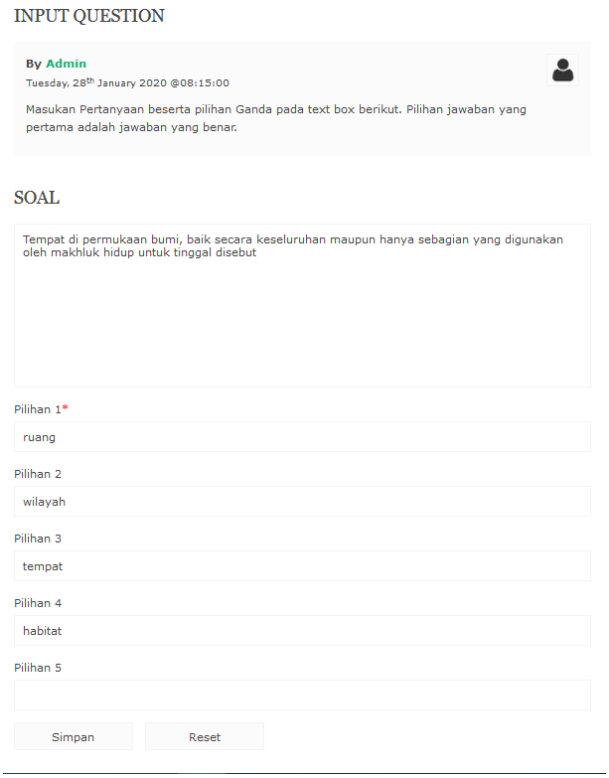
Gambar 4 di bawah ini merupakan fungsi yang digunakan dalam class *cosine* untuk menghitung nilai dari *cosine similarity*.

```
function proses(){
    //ambil text 1 :
    $str_dt1 = preg_split ("/ /", $this->str1);
    $this->dt1 = $str_dt1;
    //ambil text 2 :
    $str_dt2 = preg_split ("/ /", $this->str2);
    $this->dt2 = $str_dt2;

    //buat header T
    foreach($this->dt1 as $detail){
        if (empty(array_search($detail, $this->hd))){
            array_push($this->hd, $detail);
        }
    }
    foreach($this->dt2 as $detail1){
        if (empty(array_search($detail1, $this->hd))){
            array_push($this->hd, trim($detail1));
        }
    }

    //frekuensi vektor
    $this->isidt1 = array_count_values($this->dt1);
    $this->isidt2 = array_count_values($this->dt2);

    //membuat tabel isi dengan detail
    foreach ($this->hd as $key){
        if (array_key_exists($key, $this->isidt1) == true){
            $this->Ndt1[$key]=$this->isidt1[$key];
        }else{ $this->Ndt1[$key]=0;}
        if (array_key_exists($key, $this->isidt2) == true){
            $this->Ndt2[$key]=$this->isidt2[$key];
        }else{ $this->Ndt2[$key]=0;}
    }

    //cari nilai perkalian sekalar
    foreach ($this->hd as $key){
        $this->atas += $this->Ndt1[$key] * $this->Ndt2[$key];
    }
    //cari nilai panjang vektor
    $temp1 = 0;
    foreach($this->Ndt1 as $data_dt1){
        $temp1 += $data_dt1 * $data_dt1;
    }


    $temp2 = 0;
    foreach($this->Ndt2 as $data_dt2){
        $temp2 += $data_dt2 * $data_dt2;
    }
    $this->bawah = sqrt($temp1) * sqrt($temp2);
}
```

Gambar 4. Algoritma *cosine* dalam aplikasi

4. HASIL EKSPERIMENTAL

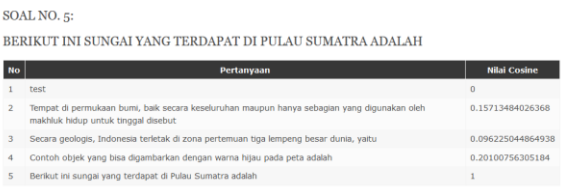
Berdasarkan eksperimantal yang dilakukan, berikut ini merupakan tampilan awal memasukan data soal dalam *database*. Gambar 5 di bawah ini merupakan contoh tampilan dalam memasukan data soal dimana *variable* yang dimasukan adalah pertanyaan, pilihan 1, pilihan 2, pilihan 3, pilihan 4, dan pilihan 5. Semua *variable* di masukan dalam *database*.



Gambar 5. Input Question

Dari data yang sudah di *input* dalam *database* seperti pada gambar 5 di atas, aplikasi kemudian akan menghitung secara otomatis dengan membandingkan soal pertanyaan yang terakhir dimasukan dengan setiap soal yang ada dalam *database*.

Pada gambar 6 di bawah adalah contoh tampilan hasil hitung *cosine* dari beberapa soal dalam *database*. Gambar 6. Menampilkan 5 soal, maka perhitungan *cosine* dihitung dengan membandingkan soal nomor 5 dengan soal nomor 1, soal nomor 5 dengan soal nomor 2, soal nomor 5 dengan nomor 3, dan soal nomor 5 dengan soal nomor 4.



Gambar 6. Input Question

Gambar 6 di atas, menampilkan nilai *cosine* dengan penjelasan sebagai berikut:

Tabel 2. Hasil Nilai *Cosine*

Soal	Nilai
Nomor 1	0
Nomor 2	0.157
Nomor 3	0.096
Nomor 4	0.201
Nomor 5	1

Nilai 0 menunjukan bahwa soal nomor 5 dengan soal nomor 1 menunjukan bahwa kesua soal tersebut sama sekali tidak sama sedangkan Nilai 1 menunjukan bahwa kedua soal yang dibandingkan adalah sama.

Jika dilihat dari nilai *cosine* dari Tabel 2 di atas, Soal nomor 4 memiliki tingkat kategori kemiripan *text* yang lebih tinggi dari pada soal nomor 3.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Aplikasi bank soal yang dibuat telah menampilkan kesamaan *text* dalam hal ini *text* yang dimaksud adalah *text* pertanyaan soal yang telah di input terakhir dalam *database* yang di bandingkan dengan *text* pertanyaan soal yang lainnya.

5.2 Saran

Bedasarkan kesimpulan yang penulis simpulkan, saran yang dapat diberikan penulis adalah sebagai berikut:

- Menentukan rentang sudut untuk mengukur kemiripan *text*.
- Menerapkan pendekatan algoritma serupa.
- Mengembangkan aplikasi dengan menerapkan feature lain.
- Membandingkan dengan algoritma yang lain.

DAFTAR PUSTAKA

[1] Khuat, Tung., Hung, Nguyen Duc,. & Hanh, Le Thi My. A Comparison of Algorithms used to measure the Similarity between two documents. April 2015. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 4 Issue 4. ISSN: 2278-1323.

[2] Rymmai, Rofeca Giri., & Saleema, JS. April 2017. Book Recommendation using Cosine Similarity. International journal of Advanced Research in Computer Science. Volume 8 No 3. ISSN No: 0976-5697.

[3] Novotný, Vít. 2018. Implementation Notes for the Soft Cosine Measure. The 27th ACM International Conference on Information and Knowledge Management. Torun, Italy: Association for Computing Machinery. pp. 1639–1642. *arXiv*:*1808.09407*. *doi*:*10.1145/3269206.3269317*. *ISBN* *978-1-4503-6014-2*.

[4] Prasetya, Didik Dwi., Wibawa, Aji., & Hirashima, Tsukasa. May 2018. The Performance of Text Similarity Algorithms. International Journal of Advances in Intelligent Informatics 4. DOI: 10.26555/ijain.v4i1.152

[5] Delphine Charlet and Geraldine Damnati. 2017. SimBow at SemEval-2017 Task 3: Soft-Cosine Semantic Similarity between Questions for Community Question Answering. In Proc. of the 11th International Workshop on Semantic Evaluation (SemEval-2017). ACL, Vancouver, Canada, 315--319

[6] Sugiyamta. 2015. Sistem Deteksi Kemiripan Dokumen dengan Algoritma Cosine Similarity dan Single Pass Clustering. Jurnal Informatika Volume 7, Nomor 2.

[7] Susandi, D. dan Sholahudin, U. 2016. Pemanfaatan Vector Space Model pada Penerapan Algoritma Nazief Adriani, KNN dan Fungsi Similarity Cosine untuk Pembobotan IDF dan WIDF pada Prototipe Sistem Klasifikasi Teks Bahasa Indonesia. Jurnal Teknologi Informasi Volume 3, Nomor 1.

[8] Nurdiana, O., Jumadi., dan Nursantika, D. 2016. Perbandingan Metode Cosine Similarity dengan Metode Jaccard Similarity pada Aplikasi Pencarian Terjemahan Al-Quran dalam Bahasa Indonesia. Jurnal Online Informatika Volume 1, Nomor 1.

[9] O. Nurdiana, J. Jumadi, and D. Nursantika, "Perbandingan Metode Cosine Similarity Dengan Metode Jaccard Similarity Pada Aplikasi Pencarian

Terjemah Al-Qur'an Dalam Bahasa Indonesia," J. Online Inform., vol. 1, no. 1, p. 59, 2016.

[10] Chaerul Hadi, M. R. M. (2017). Implementasi Cosine Similarity Dalam Aplikasi Pencarian Ayat Al-Qur'an Berbasis Android. 6(2), 71–79.