APLIKASI PENDETEKSI KEMIRIPANPADA DOKUMEN MENGGUNAKAN ALGORITMA RABIN KARP

Inta Widiastuti [1], Cahya Rahmad[2], Yuri Ariyanto[3]

[1,2]Jurusan Elektro, Program Studi Teknik Informatika, Politeknik Negeri Malang
Email : w.inta@yahoo.co.id, cahyarahmad@gmail.com, yuri.bjn@gmail.com

Abstrak

Kemiripan dokumen merupakan salah satu alternatif yang dapat digunakan untuk mengetahui penjiplakan dalam dokumen. penjiplakan adalah mencuri hasil karya orang lain dan mengakuinya sebagai karya sendiri, tanpa menyertakan referensi ke sumber aslinya. Praktik penjiplakan ini sering terjadi mengingat menyalin dokumen orang lain dapat dilakukan dengan mudah dengan proses yang bernama copy-paste. Untuk mengatasi hal tersebut dibutuhkan suatu hal yang dapat digunakan untuk membantu mendeteksi kemiripan dokumen dengan membandingkan pattern (pola) yang ada pada dokumen teks. Salah satu metode yang dapat digunakan adalah Algoritma Rabin Karp. Algoritma Rabin Karp dapat mendeteksi similarity (kemiripan) di dalam dokumen. Algoritma ini menggunakan hashing untuk menemukan substring(suatu bagian dari string) dalam sebuah teks dengan menggunakan K-gram. Fungsi hashing adalah untuk mengubah suatu jenis data menjadi beberapa bilangan bulat sederhana. Jumlah Karakter yang digunakan sebagai pattern dalam hashing disebut sebagai K-gram. Bilangan bulat hasil dari tersebut akan menjadi tolok ukur untuk mengetahui berapa similarity yang ada pada dokumen tersebut. Hasil dari penelitian ini adalah penggunaan aplikasi pendeteksi kemiripan dengan membandingkan 2 dokumen. Penelitian ini memperlihatkan bahwa algoritma ini dapat berkerja dengan baik dalam mendeteksi kemiripan dokumen dengan memperlihatkan prosentase similarity yang ada pada dokumen tersebut.

Kata kunci : Similarity,Rabin Karp ,Hashing,K-gram.

1. Pendahuluan

Seiring dengan perkembangan informasi pertukaran informasi dapat dilakukan dengan mudah. Hal tersebut tidak hanya membawa dampak positif tapi juga membawa dampak negatif. Salah satu tindakan yang dapat kita temui adalah penjiplakan. Penjiplakan bisa didefinisikan sebagai mencuri hasil karya orang lain dan mengakuinya sebagai karya sendiri, tanpa menyertakan referensi ke sumber aslinya. Praktik Penjiplakan bisa dilakukan dengan sangat mudah. Praktik penjiplakan ini sering terjadi pada kalangan mahasiswa mengingat mereka sudah familiar dengan proses yang bernama *copy-paste* yang memudahkan seseorang menyalin pekerjaan orang lain. praktik penjiplakan ini sangatlah buruk tidak hanya bagi karya orang yang dijiplak tetapi juga bagi orang yang telah melakukan penjiplakan. Kebiasaan untuk menjiplak ini dapat mematikan kreatifitas seseorang karena sudah terbiasa melakukan *copy-paste*.

Untuk membantu mendeteksi Penjiplakan, dibuatlah suatu alat yang dapat memeriksa kemiripan antar dokumen. Algorima yang digunakan oleh aplikasi ini adalah algoritma rabin karp. Algoritma ini digunakan karena efektif untuk pencarian lebih dari 1 kata (multipattern).

Proses Pendeteksian Kemiripan adalah dengan menggunakan perbandingan antara 2 dokumen yaitu dokumen yang asli dan dokumen uji. Sebelum Dokumen tersebut diolah menggunakan Algoritma Rabin Karp, Dokumen tersebut harus melalui beberapa tahap yaitu *Case Folding,Tokenezing, Filtering dan Stemming*. *Case Folding* adalah mengubah semua huruf dalam dokumen diterima dengan menghilangkan karakter selain huruf. *Tokenizing* merupakan tahapan memecah sekumpulan karakter dalam suatu teks ke dalam satuan kata. *Filtering* merupakan tahapan pengambilan kata-kata penting. Tahap stemming merupakan tahapan mencari kata dasar dari kata-kata yang

dihasilkan pada tahap filtering. Tahap *stemming* merupakan tahapan mencari kata dasar dari kata-kata yang dihasilkan pada tahap filtering. *Stemming* yang digunakan pada aplikasi ini adalah Stemming dengan menggunakan Algoritma Nazief & Andriani. Setelah melewati tahap-tahap tersebut. Dokumen tersebut akan diolah menggunakan Algoritma Rabin Karp. Algoritma ini mengubah text menjadi angka. Dan mencocokkan angka tersebut antara dokumen asli dengan dokumen uji. Hasil yang di dapatkan adalah prosentase similarity yang dihitung menggunakan Dice's Similarity Coeffient.

1.1 Landasan Teori

1.1.1 Similarity

Konsep Similarity sudah menjadi isu yang sangat penting di hampir setiap bidang ilmu pengetahuan. dalam disertasinya menjelaskan tuga macam teknik yang dibangun untuk menentukan nilai similarity (kemiripan) dokumen. (Salmuasih.2013)

a. Distance-based similarity measure
Distance-based similarity measure mengukur tingkat kesamaan dua buah objek dari segi jarak geometris dari variabel-variabel yang tercakup di dalam kedua objek tersebut. Metode Distance-based similarity ini meluputi Minkowsky Distance, Manhattan. City block distance, Euclidean distance, Jaccard Distance, Dice's Coefficient, Cosine similarity, Levenshtein Distance, Hamming Distance dan Soundex distance

b. Feature-based similarity measure
Feature-based similarity measure melakukan perhitungan tingkat kemiripan dengan merepresentasikan objek ke dalam bentuk feature-feature yang ingin dibandingkan. Feature based similarity measure banyak digunakan dalam melakukan pengklarifikasian atau pattern matching untuk gambar dan teks.

c. Probabilistic-based similarity measure
Probabilistic-based similarity measure menghitung tingkat kemiripan dua objek dengan mempresentasikan dua set objek yang dibandingkan dalam bentuk probability. Kullback Leibler Distance dan Posterior Probability termasuk dalam metode ini.

1.1.2 Algoritma Rabin Karb

Pada dasarnya, algoritma Rabin-Karp akan membandingkan nilai hash dari string masukan dan substring pada teks. Apabila sama, maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila tidak sama, maka substring akan bergeser ke kanan. Kunci utama performa algoritma ini adalah perhitungan yang efisien terhadapa nilai hash substring pada saat penggeseran dilakukan. (Firdaus Hari Bagus.2008.)

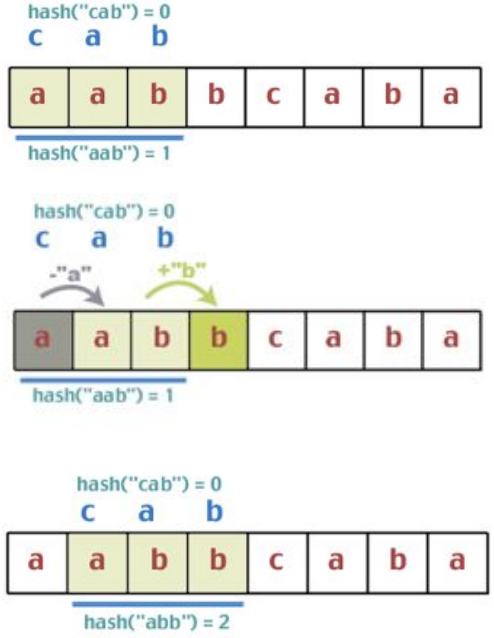
contoh cara kerja algoritma Rabin-Karp .

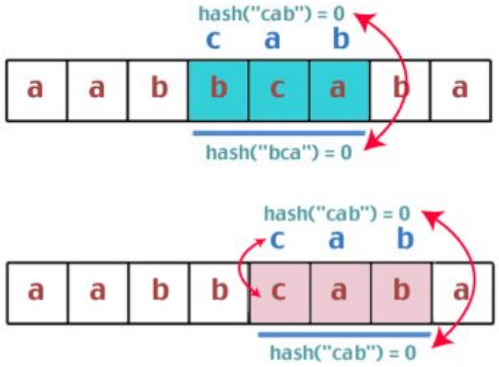
Diberikan masukan "cab" dan teks "aabbcaba". Fungsi hash yang dipakai misalnya akan menambahkan nilai keterurutan setiap huruf dalam alfabet (a = 1, b = 2, dst.) dan melakukan modulo dengan 3. Didapatkan nilai hash dari "cab" adalah 0 dan tiga karakter pertama pada teks yaitu "aab" adalah 1 seperti pada gambar 1.(Firdaus Hari Bagus.2008.)

Hasil perbandingan ternyata tidak sama, maka substring pada teks akan begeser satu karakter ke kanan. Algoritma tidak menghitung kembali nilai *hash* substring. Disinilah dilakukan apa yang disebut *rolling hash* yaitu mengurangi nilai karakter yang keluar dan menambahkan nilai karakter yang masuk sehingga didapatkan kompleksitas waktu yang relatif konstan pada setiap kali pergeseran. .(Firdaus Hari Bagus.2008.)

Berikut adalah rumus dari hashing Rabin Karp yang digunakan

$$\text{Hash} = T[i]\, b^{m-1} + T[i]\, b^{m-2} + \ldots + T[i+m-1] \bmod q$$ (Utomo Darmawan.2008)





Gambar.1 Proses Algoritma Rabin Karp

1.1.2 Hashing

Hashing adalah suatu cara untuk mentransformasi sebuah string menjadi suatu nilai yang unik dengan panjang tertentu (fixed-length) yang berfungsi sebagai penanda string tersebut. Fungsi untuk menghasilkan nilai ini disebut fungsi hash, sedangkan nilai yang dihasilkan disebut nilai hash. Contoh sederhana hashing adalah: (Firdaus Hari Bagus.2008.)

Firdaus, Hari Munir, Rinaldi

Rabin, Michael Karp, Richard

menjadi

7864 Firdaus,Hari 9802 Munir, Rinaldi

1990 Rabin, Michael 8822 Karp, Richard

Contoh diatas adalah pengunaan hashing dalam pencarian pada database. Apabila tidak di-hash, pencarian akan dilakukan karakter per karakter pada nama-nama yang panjangnya bervariasi dan ada 26 kemungkinan pada setiap karakter. Namun pencarian akan menjadi lebih mangkus setelah di-hash karena hanya akan membandingkan empat digit angka dengan cuma 10 kemungkinan setiap angka. Nilai hash pada umumnya digambarkan sebagai fingerprint yaitu suatu string pendek yang terdiri atas huruf dan angka yang terlihat acak (data biner yang ditulis dalam heksadesimal). (Firdaus Hari Bagus.2008.)

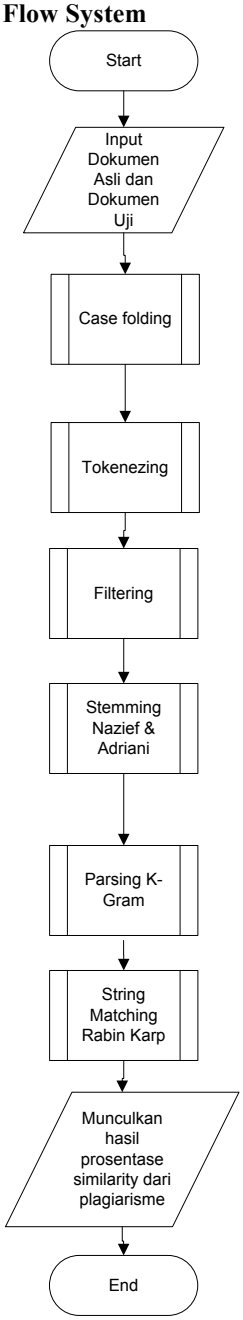
1.1.3 K-grams

Kgrams adalah rangkaian terms dengan panjang K. Kebanyakan yang digunakan sebagai terms adalah kata. K-gram merupakan sebuah metode yang diaplikasikan untuk pembangkitan kata atau karakter. Metode k-grams ini digunakan untuk mengambil potonganpotongan karakter huruf sejumlah k dari sebuah kata yang secara kontinuitas dibaca dari teks sumber hingga akhir dari dokumen. Berikut ini adalah contoh k-grams dengan k=5 : (Nugroho Eko.2011)

- Text: A do run run run, a do run run
- Kemudian dilakukanpenghilangan spasi : adorunrunrunadorunrun
- Sehingga dihasilkan rangkaian 5-grams yang diturunkan dari text : adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun.

2. Metode

2.1 Flow System



Gambar 2. Flow System Aplikasi Plagiarsime

Keterangan :

2.1.1 Case Folding

Case folding adalah mengubah semua huruf dalam dokumen diterima. Karakter selain huruf dihilangkan.

2.1.2 Tokenizing

Tahap tokenizing / parsing adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya.

2.1.3 Filtering

Filtering adalah tahap mengambil kata-kata penting dengan menggunakan algoritma stoplist (membuang kata yang kurang penting) atau wordlist (menyimpan kata penting). Stoplist/Stopword adalah kata-kata yang tidak deskriptif yang dapat dibuang. contoh dari stopwords adalah "yang", "dan",”di","dari" dan seterusnya

2.1.3 Stemming Nazief & Andriani

Stemming adalah proses mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (root word) dengan menggunakan aturan-aturan tertentu. Stemming yang digunakan oleh aplikasi ini adalah stemming yang menggunakan algoritma Nazief & Adriani . Algoritma Nazief & Adriani merupakan Algoritma stemming untuk teks berbahasa Indonesia yang memiliki kemampuan prosentase keakuratan (presisi) lebih baik dari algoritma lainnya.(Agusta Ledy.2009)

2.1.4 Rabin Karb

Algoritma Rabin-Karp adalah algoritma pencocokan string yang menggunakan fungsi hash sebagai pembanding.

2.15 Pengukuran Nilai Similarity

Pengukuran nilai similarity mengukur similarity(kemiripan) dan jarak antar dua entitas informasi adalah syarat inti pada semua kasus penemuan informasi. Penggunaan ukuran similarity yang tepat tidak hanya meningkatkan kualitas pilihan informasi tetapi juga membantu mengurangi waktu dan biaya proses. Kosinov menyarankan untuk mengaplikasikan *Dice's Similarity Coeffient* dalam perhitungan nilai similarity yang menggunakan pendekatan K-gram. (Salmuasih:2013)

$$S = \frac{K * C}{(A + B)}$$

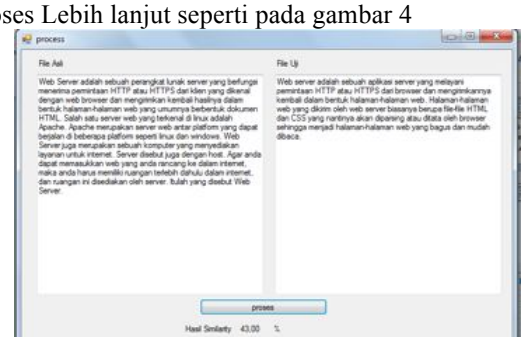
Dimana S adalah nilai similarity, A dan B adalah jumlah dari kumpulan K-gram yang sama dari teks yang dibandingkan.

3. Hasil

Berikut adalah tampilan Aplikasi

Gambar3. Tampilan Aplikasi 1

Pada Gambar3 . User diminta untuk mengisi K-Gram, Browse File Asli dan Browse File Uji. Baru kemudian File tersebut bisa di proses Lebih lanjut seperti pada gambar 4



Gambar4. Tampilan Aplikasi 2

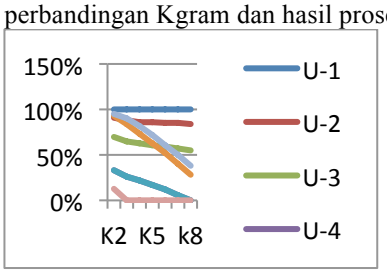
Gambar 5 adalah hasil dari perhitungan yang menunjukkan berapa persen similarity yang ada ada pada kedua dokumen tersebut

Gambar5. Hasil Prosentase Similarity

4. Pembahasan

a. Pengujian Prosentase Similarity

Terdapat 8 file yang isinya sama, hampir sama dan lain-lain. Berdasarkan data yang dikumpulkan berkut adalah grafik dari perbandingan Kgram dan hasil prosentase



Gambar8. Grafik hasil perbandingan K-gram dan hasil prosentase

Grafik tersebut menunjukkan bahwa semakin besar K-gram maka hasil prosentase similarity semakin kecil. Dan hasil data dari file U6 dan U7 dimana dokumen tersebut memiliki kata yang sama tetapi ditukar posisinya, tidak memiliki hasil yang sama dengan file yang

memiliki susunan yang sama. Hal ini membuktikan bahwa aplikasi ini mampu membedakan kata yang posisinya berbeda.

b. Pengujian menggunakan Penghitungan manual.

Terdapat 2 dokumen yang digunakan pada pengujian. Yaitu dokumen A dan dokumen U-3. Keduanya akan di hitung secara manual dan secara otomatis dari aplikasi. Kedua file akan di proses secara manual dan otomatis menggunakan tahap case Folding,tokenizing, filtering dan stemming. Kemudian kata akan di potong berdasarkan K-gram dalam hal ini k-gram yang di gunakan adalah 4. Setelah di potong kata akan di hashing. Hashing mengubah kata-kata menjadi angka seperti tabel di bawh ini.

- Pattern "plag"

Hashing = $[(16*10^{4-1})+(12*10^{4-2})+(1*10^{4-3})+(7*10^{4-4})]$ mod 26^4

$= (16000+1200+10+7) \bmod 26^4$

$= 17217 \bmod 456976$

$= 9334$

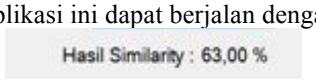
Kemudian kedua hasil hashing dari kedua file dibandingkan lalu dihitung prosentasenya similaritynya menggunakan
Dice's Similarity Coefficient.

$$S = \frac{K * C}{(A + B)}$$

S= similarity
A = jumlah kumpulan K gram pada teks 1
B = jumlah kumpulan K gram pada teks 2
C= jumlah dari K-gram yang sama dari teks yang dibandingkan,
K=2 (bigrams)

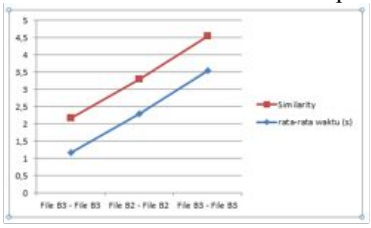
C=15 , A=33 , B=15

S= K*C/(A+B)
S=2*15/(33+15)
S= 63 %
Dengan demikian hasil manual adalah 63%. Hasil tersebut sama dengan hasil yang diproses oleh aplikasi. Ini membuktikan bahwa aplikasi ini dapat berjalan dengan baik.



c. Pengujian waktu

Dilakukan pengujian waktu menggunakan file dokumen yang memiliki isi dokumen yang berbeda. Gambar 9 merupakan grafik hasil waktu berbanding dengan waktu. Grafik ini menunjukkan bahwa semakin besar isi dokumen semakin lama waktu prosesnya



d. Pengujian isi dokumen terhadap prosentase

File Asli	File Uji	prosentase
file 1	file 1	100%
File 1	file2	4%
file 1	file 3	3%
fle1	file1 &file 2	69%
file1	file1 & File3	73%
file1	file1 &file 2& file3	57%
file1 &file 3	file1 &file 2	60%

Dengan demikian bisa disimpulkan bahwa aplikasi menghitung prosentase dari keseluruhan dokumen. Meskipun dokumen uji memiliki kalimat yang benar-benar sama dengan dokumen asli, apabila dokumen uji memiliki kalimat-kalimat lain di dalam dokumen tersebut. Hasil dari prosentase tidak akan 100%, karena aplikasi ini menghitung prosentase dari keseluruhan dokumen.

5. Kesimpulan

Dari hasil pengujian dan analisa, dapat ditarik kesimpulan untuk penelitian Aplikasi Pendeteksi Kemiripan (Similarity) menggunakan Algoritma rabin karp ini adalah:

1. Aplikasi Pendeteksi Kemiripan pada dokumen dapat dapat berjalan dengan baik.
2. Semakin besar Kgram, hasil prosentase similarity yang didapatkan semakin kecil.
3. Semakin banyak teks yang harus di proses, semakin lama waktu pemrosesannya (running time)
4. Aplikasi dapat mendeteksi kalimat dengan susunan kata yang berbeda.

5. semakin banyak kalimat yang berbeda pada tiap-tiap dokumen, hasil prosentase yang dihasilkan akan semakin sedikit, meskipun jika sebagian kalimat dari file uji sama dengan file asli.

6. Referensi

Agusta Ledy.2009. Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia.Bali: Universitas Kristen Satya Wacana

Firdaus Hari Bagus.2008. *Deteksi plagiat dokumen menggunakan algoritma rabin-karp.* Bandung : Institut Teknologi Bandung

Nugroho Eko.2011. *Perancangan sistem deteksi plagiarisme dokumen teks dengan menggunakan algoritma rabin-karp.*Malang: Universitas Brawijaya

Utomo Darmawan,Harjo Eric Wijaya dan Handoko.2008. *Perbandingan Algoritma String Searching Brute Force, Knuth Morris Pratt, Boyer Moore dan Karb Rabin pada teks Alkitab Bahasa Indonesia*.Salatiga:UKSW.

Salmuasih.2013.*Perancangan Sistem Deteksi Plagiat Pada Dokumen Teks Dengan Konsep Similarity Menggunakan Algoritma Rabin KARP*.Yogyakarta:Amikom Yogyakarta