

Penerapan Semantic Text Analysis Dan String Filter Matching Menggunakan Algoritma Boyer Moore Pada Aplikasi Chat Online

**Dadang Iskandar Mulyana*¹, Muhammad Arfan Irsyad Rowis², Ari Surya Jaya³,
Dedi Iskandar⁴**

^{1,2,3,4} Teknik Informatika, Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika, Jakarta, Indonesia

E-mail: *¹mahvin2012@gmail.com, ²arfanirsyad@gmail.com, ³arisuryajaya79@gmail.com,
⁴newwave40855@gmail.com

Abstrak

Penggunaan media sosial saat ini menjadi suatu kebutuhan sehari-hari yang tidak bisa dihindari. Berbagi pesan teks satu sama lain yang dikirimkan kepada penerima bisa saja menimbulkan suatu masalah yaitu adanya kata atau kalimat yang mengandung makna kotor ataupun tidak pantas yang ditulis oleh pengirim kepada penerima. Hal ini perlu adanya penyaringan teks untuk memberikan interaksi yang baik diantara pengguna dalam berbagi teks sebagai kebutuhan berbagi informasi. Semantic Text Analysis dalam penyaringan teks ini membandingkan kata-kata yang sama antara kata yang kotor, jika kata yang dikirim memiliki pesan yang mengandung makna tidak sopan atau kotor maka pesan teks yang dikirim kepada penerima disensor. Di dunia ilmu komputer perbandingan teks bisa diatasi dengan dengan pencarian teks. Proses pencarian pada kata yang akan dicari salah satu nya dengan menggunakan algoritma Boyer-Moore yang merupakan suatu solusi yang cukup baik untuk menyelesaikan pencarian teks secara cepat dan tepat dengan susunan karakter di dalam string yang dicocokkan memiliki jumlah atau urutan karakter di dalam string yang sama. sehingga perbandingan menjadi efisien, setelah dilakukan uji coba yang telah di lakukan masalah di atas dapat teratasi dengan baik merubah kalimat kata-kata yang tidak pantas menjadi symbol atau karakter yang di sandikan sehingga percakapan dapat berjalan dengan baik.

Kata Kunci— *Semantic Text Analysis, Penyaringan teks, Algoritma Boyer-Moore*

1. PENDAHULUAN

Keamanan Informasi merupakan salah satu aspek penting dari sebuah sistem informasi. Sayangnya masalah keamanan ini sering kali kurang mendapat perhatian dari para pemilik dan pengelola sistem informasi. Seringkali masalah keamanan berada di urutan kedua, atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting. Apabila mengganggu performansi dari sistem, seringkali keamanan dikurangi atau ditiadakan (Dowd & McHenry, 1998: 24-28) yang dikutip oleh Rahardjo [10]. Pengelolaan terhadap keamanan dapat di lihat dari sisi pengelolaan resiko (risk management). Lawrie Brown dalam (Lee, 2000) yang dikutip oleh Rahardjo [10] menyarankan menggunakan “Risk Management Model” untuk menghadapi ancaman (managing threats). Ada tiga komponen yang memberikan kontribusi kepada Risk, yaitu Asset, Vulnerabilities, dan Threats.

Sistem aplikasi chatting online yang tersedia saat ini akan meneruskan seluruh kata dan kalimat yang dikirim kepada penerima, tetapi masalah yang timbul adalah adanya kata yang kotor yang ditulis oleh pengirim juga akan diteruskan kepada penerima. Berbagi pesan menjadi suatu kebebasan dalam berinteraksi satu sama lain, tetapi penyalahgunaan teknologi bisa saja terjadi dimana pengirim bisa memberikan kata-kata tidak sopan, ancaman, phishing, kata-kata jorok dan

isu sara dikirim kepada penerima sehingga bisa menjadi pencemaran nama baik, pencemaran agama, pencemaran suku, pencemaran instansi bahkan bisa pencemaran negara, dan hal ini menjadi hal yang sangat berbahaya.

Masalah adanya kata-kata kotor dan tidak pantas yang ditulis oleh pengirim dapat diatasi dengan menggunakan algoritma Boyer-Moore yang merupakan salah dari teknik string matching. Algoritma ini merupakan jenis exact string matching algorithm yang merupakan pencocokan string secara tepat dengan susunan karakter dalam string yang dicocokkan memiliki jumlah maupun urutan karakter dalam string yang sama. Algoritma ini melakukan pencocokan dari kanan ke kiri yaitu men-scan karakter pattern dari kanan ke kiri dimulai dari karakter paling kanan. Algoritma pattern Booyer-Moore berbasis pada 2 metode yaitu The Looking Glass Technique dan The Character Jump Techique sehingga pencarian teks dapat dilakuakn lebih cepat dan efisien.

2. METODE PENELITIAN

Penggunaan kriptografi pada aplikasi chatting online bertujuan untuk menyaring pesan teks yang mengandung kata-kata kotor yang dikirim satu sama lain walaupun berada dalam jalur komuniksai yang sama dengan pihak lain. Mihir Bellare & Phillip Rogaway dalam bukunya mendefinisikan Kriptografi, "Historically, cryptography arose as a means to enable parties to maintain privacy of the information they send to each other, even in the presence of an adversary with access to the communication. [7]

Menurut Schneier (1996), Menezes (1996), dan Wikipedia (2006) yang dikutip oleh Munir [8], selain berdasarkan sejarah yang membagi kriptografi menjadi kriptografi klasik dan modern. Maka berdasarkan kunci untuk enkripsi dan deskripsi, kriptografi dapat dibedakan menjadi kriptografi kunci simetri dan kriptografi kunci nirsimetri dan keduanya termasuk dalam kriptografi modern. Sedangkan menurut Ariyus yang dikutip dari Jurnal Dadang [9] berdasarkan kunci untuk enkripsi dan deskripsi selain kriptografi kunci simetri dankriptografi kunci nirsimetri ada juga fungsi hash. Algoritma kriptografi merupakan langkah- langkah logis bagaimana menyaring pesan dari kata-kata yang tidak sepatasnya diucapkan atas pesan tersebut dengan melakukan pembangkitan kunci, enkripsi dan dekripsi. Kriptografi bertujuan untuk memberi layanan keamanan (yang juga dinamakan sebagai aspek-aspek keamanan).

2.1. Aspek Kriptografi

Dikutip dari jurnal M. Arif & Dadang Iskandar [11] Tujuan kriptografi adalah memberikan keamanan. Dalam memberikan layanan keamanan kriptografi harus meliputi aspek-aspek :

1. Authority, menjaga informasi dari pihak yang tidak memiliki otoritas.
2. Integrity, bahwa informasi yang di terima tidak berubah dan sesuai dengan aslinya.
3. Authentication, pengenalan pada pengguna yang saling berhubungan dan identifikasi kebenaran informasi.
4. Nonrepudation, bahwa penerima dan pengirim informasi tidak bisa memberikan penyangkalan atas informasi yang telah diterimanya.

String matching merupakan algoritma pencocokan string. Sifat algoritma string matching adalah mencari sebuah string yang terdiri dari beberapa karakter (yang biasa disebut pattern). merupakan bagian dari pemrosesan data, pengkompresian data, lexical analysis, dan temu balik informasi. Teknik untuk menyelesaikan permasalahan pencocokan string biasanya akan menghasilkan implikasi langsung ke aplikasi string lainnya. Pencocokan string atau string

matching adalah proses pencarian semua kemunculan string sehingga string matching adalah algoritma yang ditujukan untuk melakukan pencocokan sub string pada string besar [1].

String matching adalah pencarian sebuah pattern pada sebuah teks atau dengan kata lain sebuah pencarian pola string yang lebih pendek didalam string yang lebih panjang. String matching atau pencocokan merupakan bagian penting dari sebuah proses pencarian. Penyaringan kata merupakan cara untuk mempermudah pengguna dalam menyaring kata-kata kotor yang mereka inginkan.

Adapun algoritma String Matching dibagi menjadi tiga kategori menurut arah pencariannya :

Dari arah kiri ke kanan, atau bisa disebut dari awal ke akhir :

1. Algoritma Brute Force
2. Algoritma dari Morris dan Pratt, yang kemudian dikembangkan oleh Knuth, Morris, dan Pratt

Dari arah kanan ke kiri :

1. Algoritma dari Boyer dan Moore, yang kemudian banyak dikembangkan, menjadi Algoritma turbo Boyer-Moore, Algoritma tuned Boyer-Moore, dan Algoritma Zhu-Takaoka

Dan kategori terakhir, dari arah yang ditentukan secara spesifik oleh algoritma tersebut, arah ini menghasilkan hasil terbaik secara teoritis, algoritma yang termasuk kategori ini adalah:

1. Algoritma Colussi
2. Algoritma Crochemore-Perrin

Dalam suatu kasus diberikan misal teks dengan x sebagai karakter dalam array $A[1...x]$ dan suatu pola lain yaitu dengan Y sebagai karakter dalam array $B[1...Y]$. bagaimana mendapatkan integer s yang disebut dengan valid shift? Terlebih dahulu valid shift ini memenuhi kriteria sebagai berikut dimana $0 \leq s < x - y$ dan $A[s + 1 \dots s + y] = B[1 \dots y]$. Jika B sebagai substring dari A , maka selanjutnya akan diketahui apakah B didalam A . Elemen-elemen B dan A dapat berupa karakter atau alfabet seperti $\{0,1\}$ atau $\{A,B,C,D,E,\dots,Z,a,b,c,d,\dots,z\}$. Untuk kasus yang membutuhkan string matching dengan efisiensi tinggi dari aplikasi, dapat menggunakan algoritma Boyer Moore. Contoh penggunaan algoritma ini adalah untuk menyunting suatu teks, dan substitusi komentar kedalam teks. Algoritma ini memungkinkan string matching dengan ukuran alphabet moderat dan pola teks yang relatif panjang dalam kecepatan tinggi. Selama pengujian ada kemungkinan penempatan pola A dalam data teks B . jika karakter $B[i] = c$ tidak sebanding dengan pola $A[j]$ maka akan berjalan ketentuan tertentu : 1. Jika c tidak berisi dan dalam A boleh dimana saja, maka akan geser pola A melalui $T[i]$ 2. Disaat kondisi lain, geser A sampai karakter c sehingga A dapat lurus dengan $B[i]$. Teknik ini agar menghindari terjadinya tumpukan antara perbandingan pola pergeseran relatif terhadap data teks. Dapat disimpulkan bahwa algoritma boyer moore relatif lebih cepat pada alfabet besar (panjang pola teks), hasilnya bukan untuk string biner atau membentuk pola pendek, untuk membuat string biner lebih diutamakan menggunakan algoritma knuth-morris-pratt. Sedangkan untuk pola-pola yang pendek dapat menggunakan algoritma naïve karena saat eksekusi berlangsung worst case akan berbentuk quadratic[2].

String matching secara garis besar bisa dibedakan menjadi dua konsep yaitu sebagai berikut:

1. Exact string matching, merupakan pencarian string secara tepat dengan menyusun karakter dalam string yang memiliki kecocokan jumlah maupun urutan karakter di dalam string yang sama. Sebagai contoh : close maka akan menunjukkan hanya dengan kata close [3].
2. Inexact string matching, merupakan pencarian string dimana string akan dicocokkan yang memiliki kemiripan diaman keduanya memiliki susunan karakter dari kata yang berbeda tetapi

string tersebut memiliki kemiripan baik dari penulisan/tekstual (approximate string matching) atau kemiripan sebuah ucapan (phonetic string matching) [3].

2.2. Algoritma Boyer Moore

Algoritma Boyer-Moore diperkenalkan oleh Bob Boyer dan J.S. Moore pada tahun 1977. Pada algoritma ini pencocokan kata dimulai dari karakter terakhir kata kunci menuju karakter awalnya. Jika terjadi perbedaan antara karakter terakhir kata kunci dengan kata yang dicocokkan maka karakter-karakter dalam potongan kata yang dicocokkan tadi akan diperiksa satu per satu. Hal ini dimaksudkan untuk mendeteksi apakah ada karakter dalam potongan kata tersebut yang sama dengan karakter yang ada pada kata kunci. Apabila terdapat kesamaan, maka kata kunci akan digeser sedemikian rupa sehingga posisi karakter yang sama terletak sejajar, dan kemudian dilakukan kembali pencocokan karakter terakhir dari kata kunci. Sebaliknya jika tidak terdapat kesamaan karakter, maka seluruh karakter kata kunci akan bergeser ke kanan sebanyak m karakter, di mana m adalah panjang karakter dari kata kunci. Algoritma Boyer-Moore adalah pencocokan pattern dari kanan ke kiri maka, informasi yang didapat akan lebih banyak, seperti pattern dengan panjang pattern diletakkan pada ujung kiri atas dari sebuah String, sehingga kedua karakter yang pertama sejajar dan char adalah karakter ke-pattern dari string, maka karakter tersebut akan sejajar dengan karakter terakhir dari pattern. Algoritma Boyer-Moore telah banyak dikenal oleh masyarakat dan dianggap paling efisien untuk pencarian string. Ide di balik algoritma ini adalah dengan memulai pencocokan karakter dari kanan dan bukan dari kiri, maka akan lebih banyak informasi yang didapat [4][5].

2.3. Tahapan Algoritma Boyer Moore

Langkah-langkah yang digunakan pada algoritma Boyer-Moore pada saat mencocokkan string di bagi menjadi dua yaitu sebagai berikut:

1. Good suffix heuristic

Mengambil contoh ketika t adalah substring dari teks T yang dicocokkan dengan substring dari pola P . Sehingga akan digeser polanya sampai kemunculan lain dari t di P yang dicocokkan dengan t di T , kemudian sebuah awalan dari P , yang cocok dengan akhiran dari t , dan terakhir P bergerak melewati t . Untuk menggunakan Good suffix heuristic yaitu sebagai berikut :

Kasus 1: Kemunculan pattern lain dari t di P yang dicocokkan dengan t di T

Text: ABAABABACBA

Pattern: CABAB

Langkah Untuk : Pencocokan pertama Good suffix heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	B	B	A	A	B	A	B	A	C	B	A
P	C	A	B	A	B						

Langkah Untuk : Penyelesaian dan pencocokan kedua Good suffix heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	B	B	A	A	B	A	B	A	C	B	A
P			C	A	B	A	B				

tahapan di atas, kita telah mendapatkan *substring* t dari teks T yang cocok dengan pola P (berwarna hijau) sebelum mismatch pada indeks 2. Sekarang kita akan mencari kemunculan t ("AB") di P . Kita telah menemukan sebuah kemunculannya dimulai dari posisi 1 (berlatar

kuning) jadi kita akan menggeser pola ke kanan 2 kali untuk menyelaraskan t di P dengan t di T.

Kasus 2: P bergerak melewati t

Text: AACABABACBA

Pattern: CBAAB

Langkah Untuk: Pencocokan pertama Good suffix heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	D	A	C	A	B	A	B	A	C	B	A
P	C	B	A	A	B						

Langkah Untuk : Penyelesaian dan pencocokan kedua Good suffix heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	D	A	C	A	B	A	B	A	C	B	A
P						C	B	A	A	B	

Tahapan di atas, tidak ada kemunculan t ("AB") di P dan juga tidak ada awalan di P yang cocok dengan akhiran t. Jadi, dalam hal ini, kita tidak akan pernah menemukan kecocokan sempurna sebelum indeks 4, jadi kita akan menggeser P melewati t yaitu. untuk indeks 5.

Kasus 3: Awalan P, yang cocok dengan akhiran t di T

Langkah Untuk : Pencocokan pertama Good suffix heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	A	A	B	A	B	A	B	A	D	C	A
P	A	B	B	A	B						

Langkah Untuk : Penyelesaian dan pencocokan kedua Good suffix heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	A	A	B	A	B	A	B	A	D	C	A
P					A						

Tahapan diatas, kita mendapatkan t ("BAB") yang dicocokkan dengan P (berwarna hijau) pada indeks 2-4 sebelum mismatch. Tetapi karena tidak ada kemunculan t di P, maka akan mencari beberapa awalan P yang cocok dengan beberapa akhiran t. Dimana telah menemukan awalan "AB" (di latar belakang kuning) mulai dari indeks 0 yang tidak cocok dengan seluruh t tetapi akhiran t "AB" mulai dari indeks 3. Jadi sekarang akan menggeser pattern 3 kali untuk menyelaraskan awalan dengan akhiran.

2. Bad Character Heuristic

Bad Character Heuristic untuk karakter teks yang tidak sesuai dengan karakter pada pattern saat ini dan disebut sebagai Bad Character. Ketika tidak adanya kecocokan, maka akan menggeser pattern hingga

1. Pattern P bergerak melewati karakter yang tidak cocok.
2. Ketidakcocokan menjadi kecocokan

Tahap menggunakan Bad Character Heuristic dalam pencarian teks yaitu sebagai berikut:

Kasus 1: Mismatch menjadi match

Text: BCAATGCCT

Pattern: TATGTG

Langkah Untuk : Pencocokan pertama Bad character heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	B	C	A	A	T	G	C	C	T	A	T
P	T	A	T	G	T	G					

Langkah Untuk : Penyelesaian dan pencocokan kedua Bad character heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	B	C	A	A	T	G	C	C	T	A	T
P		T	A	T	G	T	G				

Pada tahap diatas, mendapatkan mismatch pada posisi 3. Disini karakter mismatch adalah "A".

Sekarang akan mencari kemunculan terakhir dari "A" dalam pattern. Dalam teks mendapat "A" di posisi 1 dalam pattern (ditampilkan dengan warna Biru) dan ini adalah kemunculan terakhirnya. Sekarang akan menggeser pola 2 kali sehingga "A" dalam pola sejajar dengan "A" dalam teks.

Kasus 2: Pattern bergerak melewati karakter yang tidak cocok

Text: CAATGCCTATG

Pattern: TATG

Langkah Untuk : Pencocokan pertama Bad character heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	C	A	A	T	G	C	C	T	A	T	G
P				T	A	T	G				

Langkah Untuk : Penyelesaian dan pencocokan kedua Bad character heuristic

I	0	1	2	3	4	5	6	7	8	9	10
T	C	A	A	T	G	C	C	T	A	T	G
P								T	A	T	G

Tahap diatas, Di sini kita memiliki ketidaksesuaian pada posisi 6. Karakter ketidakcocokan "C" tidak ada dalam pola sebelum posisi 6 jadi kita akan menggeser pola melewati ke posisi 6 dan akhirnya pada contoh di atas telah mendapatkan kecocokan pola yang sempurna (ditampilkan dalam warna Orange). dengan melakukan ini karena "C" tidak ada dalam pola sehingga pada setiap shift sebelum posisi 6 akan mendapatkan ketidaksesuaian dan pencarian akan sia-sia.

2.4. Kelebihan dan Kelemahan Algoritma Boyer Moore

Boyer-Moore membandingkan karakter dari kanan ke kiri dan memiliki loncatan karakter yang besar sehingga mempercepat pencarian string karena dengan hanya memeriksa sedikit karakter, dapat langsung diketahui bahwa string yang dicari tidak ditemukan dan dapat digeser ke posisi berikutnya. kelemahan algoritma BoyerMoore mencocokkan pattern dari kanan ke kiri oleh sebab itu kelemahan dari algoritma ini adalah ketika semua karakter memiliki kesamaan atau cocok dan hanya karakter terakhir atau karakter paling kiri yang berbeda maka pencarian ini akan memerlukan waktu yang sedikit lama.

Tabel 1. Contoh algoritma Boyer-Moore

Teks	G	R	A	C	E			
Pattern	H	A	L	I	M			

Pada tabel 1, dengan melakukan pencocokan dari posisi paling akhir/kanan pattern dapat dilihat bahwa karakter “M” pada pattern “HALIM” tidak cocok dengan karakter “E” pada teks “GRACE”, dan karakter “E” tidak pernah ada dalam pattern “HALIM” yang dicari sehingga pattern “HALIM” dapat digeser melewati teks “GRACE” sehingga posisinya menjadi:

Tabel 2, Contoh pergeseran algoritma BoyerMoore

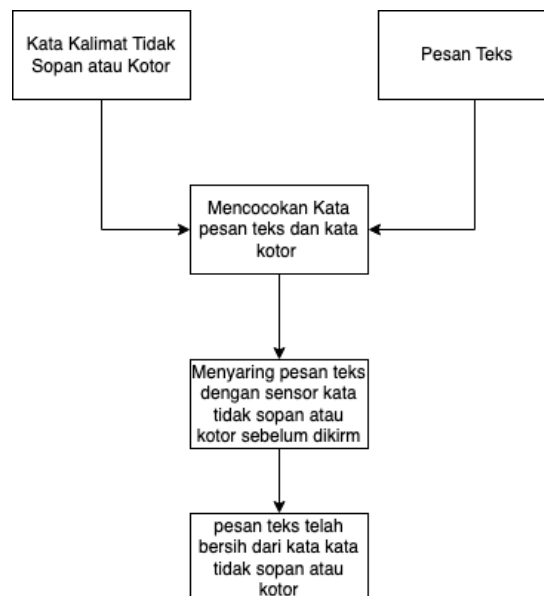
Teks	G	R	A	C	E					
Pattern						H	A	L	I	M

Tabel 2, menunjukkan bahwa algoritma BoyerMoore memiliki pergeseran karakter yang besar sehingga mempercepat pencarian pattern karena dengan hanya memeriksa sedikit karakter, dapat langsung diketahui bahwa pattern yang dicari tidak ditemukan dan dapat digeser ke posisi berikutnya. Algoritma Boyer-Moore menggunakan dua buah tabel untuk mengolah informasi saat terjadi kegagalan pencocokan pattern. Tabel pertama disebut bad character shift juga sering disebut occurrence heuristic (OH). Tabel kedua disebut dengan istilah good suffix shift juga disebut match heuristic (MH)[6].

3. HASIL DAN PEMBAHASAN

3.1. Kata dan Penyaringan Teks

Kata dan pesan teks yang dimasukan merupakan kalimat bebas saat diinput untuk nantinya digunakan untuk berbagi informasi kepada penerima, adapun kata-kata kurang sopan dan kotor yang diinput oleh pengguna akan disaring di dalam sebuah database penyimpanan yang di dalamnya terdapat list kata dan pattern yang telah ditetapkan untuk di saring



Gambar 1. Proses penyaringan teks pada kata tidak sopan atau kotor

Pada alur proses diatas mengalami beberapa proses. Kata yang mengandung makna tidak sopan atau kotor akan sudah ada di dalam penyimpanan data yang besar yang nantinya

akan terus bertambah di kemudian hari. Semakin banyak kata tidak sopan atau kotor maka oleh sistem akan dilakukan pencarian pesan teks dengan membandingkan antara kata setiap pesan kalimat dengan melakukan perulangan. Kata tidak sopan atau kotor terdapat beberapa kategori seperti kategori cacat, kategori kebodohan, kategori kelamin, kategori umum, kategori binatang.

3.2. Penyaringan Teks

Menentukan kata tidak sopan atau kotor sebagai pattern. Penyaringan teks membutuhkan sedikitnya dua buah parameter. Parameter pertama berisi teks pesan kalimat dan parameter kedua berisi patter atau pola yang nantinya akan disandingkan dengan parameter pertama dari karakter per-karakter.

Keterangan :

P = *Pattern* = pola kata

T = *Text* = kumpulan kata atau kalimat

t = substring *Text*

p = substring *Pattern*

Langkah Untuk : Penyelesaian pencarian string pada kata “BIADAP”

I	0	1	2	3	4	5	6	7	8	9	10
T	K	A	M	U		B	I	A	D	A	P
P	B	I	A	D	A	P					
P						B	I	A	D	A	P

Berikut penjelasan dari proses table diatas untuk mudah dalam memahami berdasarkan pencarian kata yang akurat:

1. Dimana proses penyaringan kata ini berjalan dari samping kanan kiri.
2. Pencocokan pertama kali, pattern terdapat pada indeks ke-0 dengan jumlah panjang karakter bernilai 6. “B” yang berada pada indeks ke-5 idak cocok dengan pattern yang terdapat di dalam pattern paling kanan yakni “P”. Ketika tidak adanya kecocokan, maka karakter terakhir yang sejajar dengan pattern paling akan akan mencari karakter di dalam pattern. Ketika di dalam pattern berisi karakter yang dimaksud “B” maka akan menggeser sebanyak 5 kali sehingga selaras dengan karakter Text.
3. Pencocokan kedua kali, setelah dilakukan pengecekan terhadap pattern dan teks dari kanan ke kiri, ditemukan kecocokan seluruh pattern pada teks. Keocokan seluruh pattern telah berhasil dan pencarian teks untuk filtering bisa dilakukan.

3.3. Perancangan Database

Di dalam sistem yang akan dibuat terdapat 3 tabel perancangan database. Berguna untuk menyimpan data yang akan di masukan oleh program aplikasi nantinya. Tabel tersebut diantaranya yaitu:

Tabel 1. Struktur tabel Corpus

Kolom	Tipe	Ukuran	Keterangan
Id	Varchar	10	Primary Key
Kata	Varchar	200	
Kategori	Varchar	30	
Status	Varchar	10	

Tabel 2. Struktur tabel Pesan

Kolom	Tipe	Ukuran	Keterangan
Id	Int	10	Primary Key
Waktu	datetime		
Pengirim	Varchar	200	
Penerima	Varchar	30	
Pesan	Varchar	10	

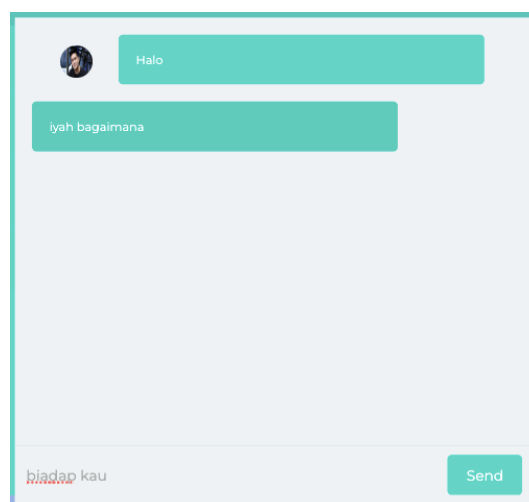
Tabel 3. Struktur tabel User

Kolom	Tipe	Ukuran	Keterangan
Id	Int	10	Primary Key
Email	Varchar	20	
Password	Varchar	16	
Status	Varchar	20	
Email pemulihan	Varchar	15	

Tabel kata tidak sopan atau kotor digunakan untuk menyimpan daftar kata yang nantinya akan dijadikan sebagai pattern di penyaringan kata atau kalimat, dengan diberikan kategori berdasarkan kata yang disaring nantinya. Tabel pesan digunakan untuk menyimpan informasi pengirim dan juga penerima dengan maksimal pesan 255 karakter di dalamnya, dengan menyimpan history waktu yang nantinya ditampilkan. Tabel user digunakan untuk menyimpan informasi data user seperti email dan sandi yang bisa digunakan untuk login kedalam aplikasi. Dengan penerapan seperti ini bisa memberikan keamanan lebih dalam berbagi informasi kepada penerima yang dituju.

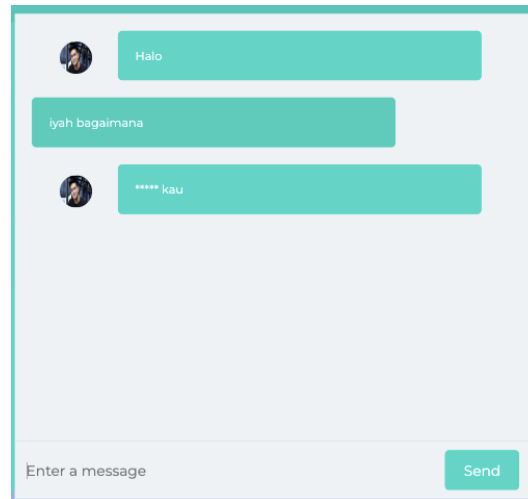
3.4. Implementasi

Implementasi merupakan tahap lanjut setelah adanya perancangan sistem, pada implementasi dijelaskan mengenai bagaimana dan mengoperasikan aplikasi atau sistem penyaringan teks yang telah dibangun oleh algoritma Boyer-Moore untuk penyaringan teks.



Gambar 2. Proses pesan teks pengirim kepada penerima

Tampilan diatas muncul apabila user telah masuk kedalam room sang penerima dengan mendaftar terlebih dahulu dan saling bertukar id agar bisa saling terhubung. Sebagai contoh pada tampilan diatas, sang pengirim akan mengirim kalimat tidak sopan atau kotor kepada penerima. Setelah pengirim menekan tombol send maka pesan akan dikirim ke server dengan adanya proses penyaringan teks.



Gambar 3. Hasil penyaringan string menggunakan Algoritma Boyer-Moore

Ketika pengirim telah menekan tombol send. Dimana kata yang mengandung makna tidak sopan atau kotor akan di filter kemudian disensor oleh sistem secara otomatis dengan menggunakan algoritma Boyer-Moore. Proses ini terjadi ketika pesan akan disimpan kedalam server sehingga sebelum kata kotor sampai langsung kepada penerima, sudah adanya string filtering dan string matching terlebih dahulu.

4. KESIMPULAN

Algoritma Boyer-Moore dapat digunakan sebagai penyaringan teks dan juga pencocokan teks guna melakukan proses Semantic Text Analysis didalam mengolah kata dan membandingkan dengan pattern yang telah ditentukan. Ketika algoritma ini digunakan di dalam aplikasi chat online, bisa memberikan dampak yang sangat baik karena bisa memfilter kata yang tidak sopan atau kotor dengan melakukan sensor atau menghapus kata kotor yang dimaksud. Hasil uji coba yang telah dilakukan bahwa efektifitas Algoritma Boyer-Moore terbilang cepat dalam proses komputasi yang dilakukannya.

DAFTAR PUSTAKA

- [1] Tomy Satria Alasi, "Algoritma Boyer Moore Untuk Penyaringan Pesan Teks Menggunakan Perbandingan Kata Yang Sama," *Semin. Nas. Sains Teknol. Inf.*, no. April 2017, pp. 488–495, 2018.
- [2] D. S. , D. T. , Uray Ristian, "Penerapan Algoritma Boyer Moore Dan Metode N-Gram Pada Aplikasi Penyunting Naskah Teks Bahasa Indonesia Berbasis Web," *Coding J. Komput. dan Apl.*, vol. 8, no. 3, p. 50, 2020, doi: 10.26418/coding.v8i3.42957.
- [3] R. I. Borman, "Penerapan String Matching Dengan Algoritma Boyer Moore Pada Aplikasi

- Font Italic Untuk Deteksi Kata Asing,” *J. Teknoinfo*, vol. 10, no. 2, p. 39, 2016, doi: 10.33365/jti.v10i2.9.
- [4] E. Rahmanita, “Pencarian String Menggunakan Algoritma Boyer Moore Pada Dokumen,” *J. Ilm. NERO*, vol. 1, no. 1, pp. 15–26, 2014.
- [5] W. S. Saputri, R. Dhenabayu, F. Febrinita, P. Studi, and T. Informatika, “Copyrights @ Balitar Islamic University , Blitar-Indonesia APPLICATION OF STRING MATCHING USING BOYER MOORE ALGORITHM IN THE APPLICATION OF DICTIONARY Copyrights @ Balitar Islamic University , Blitar-Indonesia,” vol. 1, no. 1, pp. 11–20, 2018.
- [6] C. Irawan, M. R. Pratama, V. Wahanggara, T. Informatika, and U. Muhammadiyah, “PERBANDINGAN ALGORITMA BOYER MOORE DAN BRUTE FORCE PADA PENCARIAN KAMUS BESAR BAHASA INDONESIA BERBASIS ANDROID”.1977.
- [7] Mihir Bellare, Phillip Rogaway. *Introduction To Modern Cryptography*. ©Mihir Bellare and Phillip Rogaway.1997–2005.
- [8] Munir, R. (2006) Kriptografi. Bandung:Informatika
- [9] Dadang Iskandar Mulyana, “KAJIAN PENERAPAN ENCODE DATA DENGAN BASE64 PADA PEMROGRAMAN PHP,” *Jurnal CKI On SPOT*, Vol. 9, No. 1, JUNI 2016.
- [10] Rahardjo, B. (2005), “Keamanan Sistem Informasi Berbasis Internet” Jakarta: PT. INDOCISC.
- [11] Muhammad Arif Zainuddin, Dadang Iskandar Mulyana, “Penerapan Algoritma RSA untuk keamanan pesan instan pada perangkat android” *Jurnal CKI On SPOT*, Vol.9, No.2, Desember 2016.