REPOSITOR, Vol. 2, No. 11, November 2020, Pp. 1521-1528
ISSN : 2714-7975
E-ISSN : 2716-1382

Implementasi Algoritma Graf dan Algoritma Genetika pada Peringkasan Single Document

Lina Dwi Yulianti[1], Setio Basuki[2], Yufis Azhar[3]
Universitas Muhammadiyah Malang
e-mail: linadwiy@gmail.com[1], setio_basuki@umm.ac.id[2], yufis@umm.ac.id[3]

Abstrak

Dalam kemajuan teknologi sekarang ini, pencarian sebuah informasi lebih mudah dan cepat. Namun tidak sedikit informasi yang tidak benar adanya atau biasanya disebut dengan istilah hoax. Maka dari itu, informasi harus didapatkan dari beberapa sumber untuk memastikan keakuratan dari suatu informasi. Sistem Automatic Text Summarization adalah suatu sistem yang digunakan untuk proses peringkasan dokumen yang berbasis text. Sistem ini dapat membantu menemukan inti dari sebuah dokumen berita, sehingga tidak memerlukan banyak waktu untuk membaca. Dalam penelitian ini digunakan Algoritma Graf dan Algoritma Genetika dalam pembangunan sistem. Dari hasil pengujian, didapatkan keakurasian antara ringkasan yang dihasilkan oleh sistem dengan ringkasan manual memiliki nilai Cosine similarity sebesar 71,21%. Hal tersebut dapat menunjukkan jika sistem yang dibangun dapat digunakan oleh pengguna karena hasil pengujian yang dilakukan mendapatkan nilai keakurasian yang cukup tinggi.

Kata kunci: *Text Summarization, Algoritma Graf, Algoritma Genetika.*

Abstract

In today's technological advancements, finding information is easier and faster. But not a little information that is not true or commonly referred to as hoaxes. Therefore, information must be obtained from several sources to ensure the accuracy of the information. Automatic Text Summarization System is a system used for text based document summarization. This system can help find the core of a news document, so it does not require much time to read. Researchers use Graph Algorithms and Genetic Algorithms in system development. From the test results obtained by the accuracy of the system produced by the system with manual numbers have a cosine similarity value of 71.21%. This can prove that the system built can be used by users because the results of tests carried out get high accuracy values.

Keywords*: Text Summarization, Graph Algorithms, Genetic Algorithms.*

1. Pendahuluan

Dalam kemajuan teknologi sekarang ini suatu informasi tidak didapatkan hanya melalui koran ataupun buku saja. Pada saat ini informasi bisa kita dapatkan melalui media elektronik seperti *smartphone* laptop, dll, dengan memanfaatkan koneksi jaringan internet. Untuk mengetahui suatu informasi dalam negeri atapun luar negeri dapat dengan mudah dibaca pada *smartphone* ataupun media lain selama terhubung dengan koneksi internet. Pencarian suatu informasi dapat dilakukan dimana dan kapan saja. Namun tidak sedikit juga yang memanfaatkan media elektronik untuk menuliskan informasi – informasi yang tidak benar adanya atau yang biasa disebut dengan *hoax*.

Dalam mencari suatu informasi tidak hanya dari satu sumber saja, jika menginginkan informasi yang lebih akurat, kita harus mencari informasi dari beberapa sumber. Untuk menemukan ide pokok dari setiap sumber, kita harus membaca artikel satu per satu dan menggabungkan hasil ringkasan ke dalam satu kesimpulan. Hal seperti itu akan membutuhkan waktu yang sangat lama jika dilakukan secara manual. Dengan penjabaran permasalahan diatas, sistem *Automatic Text Summarization* diciptakan untuk memberikan solusi.

Automatic Text Summarization adalah suatu sistem yang digunakan untuk proses peringkasan dokumen yang berbasis text [1], berdasarkan jumlah dokumen yang diproses,

peringkasan dokumen dapat dikategorikan menjadi dua yaitu peringkasan dokumen tunggal dan *multi-dokumen*[2]. Dalam penelitian ini membahas tentang peringkasan teks menggunakan *single dokumen*, dan algoritma yang digunakan adalah algoritma genetika. Algoritma Genetika adalah suatu metode untuk mencari suatu lintasan terpendek pada sebuah dokumen (*shortest path finding*)[3].

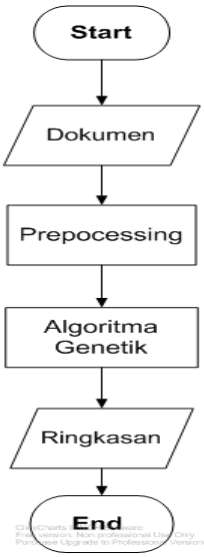
Terdapat beberapa jurnal sebelumnya tentang *summarization* baik menggunakan dokumen tunggal maupun *multi dokumen*. Penelitian [3] menggunakan metode algoritma graf dan algoritma genetika, penelitian ini membahas tentang perhitungan manual dan kesesuaian hasil perhitungan pada sistem, pengujian waktu proses serta pengujian sumber daya. Dari penelitian ini dapat disimpulkan bahwa nilai generasi dan populasi berbanding lurus dengan waktu proses dan penggunaan sumber daya.

Pada penelitian [4] sebuah implementasi terkait dengan automasi peringkasan teks bertipe ekstraktif dengan menggunakan metode TF-IDF-EGA. EGA (*Enhanced Genetic Algoritm*) merupakan metode pengembangan dari *Genetic Algoritm*(GA). Alur dari algoritma ini mirip dengan algoritma genetika, perbedaannya terletak pada operator reproduksi mutasinya. Operator reproduksi mutasi ini dikembangkan lagi untuk mengatasi randomisasi nilai *string chromosome* yang berefek pada keragaman populasi. Dari pengujian yang dilakukan, menyimpulkan bahwa parameter yang dapat menghasilkan nilai *fitness* terbaik adalah dengan jumlah iterasi sebesar 50, jumlah populasi sebesar 30, nilai *crossover rate* sebesar 0.3, dan nilai *mutation rate* sebesar 0.8. Dimana semakin tinggi nilai iterasi dan populasi dengan parameter yang tidak terlalu jauh akan semakin meningkatkan rata-rata nilai *fitness*, begitu pula untuk parameter reproduksi.

Pada penelitian sebelumnya [3] dilakukan pembobotan dengan penentuan posisi kalimat pada suatu paragraf. Pada kalimat awal dan akhir sebuah paragraf akan diberikan nilai yang lebih tinggi dari kalimat yang lainnya. Penelitian ini tidak menggunakan pembobotan posisi kalimat tersebut karena dalam dokumen berita, awal dan akhir kalimat pada suatu paragraf belum tentu sebagai ide pokok suatu paragraf, dan isi dokumen berita tidak sebanyak artikel lainnya. Namun pada penelitian ini dilakukan pembobotan kalimat yang mirip dengan judul (*topic relation factor*),pembobotan untuk menentukan apakah kalimat dalam sebuah ringkasan berisikan informasi yang sama atau tidak (*cohesion factor*), pembobotan untuk memudahkan pembacaan pada sebuah ringkasan (*readibility factor*). Implementasi penelitian ini menggunakan algoritma graf dan algoritma genetika untuk peringkasan dokumen berita dalam bahasa pemograman *phyton*.

2. Metode Penelitian

Sebelum membangun sebuah sistem ATS yang menggunakan metode yang diusulkan, penulis membuat analisis dan rancangan dari sistem. Secara garis besar sistem yang akan dibangun memiliki alur sebagai berikut:



Gambar 1. Alur sistem ATS

Gambar 1 merupakan gambaran umum tentang proses pada sistem ATS yang akan dibangun, untuk penjelasan alur akan dijabarkan sebagai berikut :

2.1 Data

Pengambilan data untuk penelitian ini didapatkan dari internet yaitu pada situs detik.com,okezone.com,dan liputan6.com. Ketegori berita yang diambil yaitu tentang bencana alam yang terjadi di Indonesia pada tahun 2019-2020 dan panjang berita minimal 200 kata. Data yang digunakan untuk melakukan pengujian sistem sebanyak 10 dokumen berita.

2.2 Preprocessing Data

Pada preprocessing data terdapat beberapa tahap untuk mengubah dokumen yang tidak terstruktur menjadi dokumen yang terstuktur. Tahap – tahap dalam melakukan preprocessing dijelaskan sebagai berikut :

1. Tokenization

Tokenization adalah proses untuk menguraikan teks dokumen menjadi kalimat, dan menguraikan kalimat menjadi per kata – kata[5]. Pada proses ini dokumen akan dipecah menjadi perkalimat. Dari kalimat tersebut akan dipecah lagi menjadi per kata. Keluaran dari proses ini adalah daftar semua kata – kata yang ada dalam dokumen. Daftar kata akan disimpan dalam basis daftar kata[5].

2. Stop Word

Stop word adalah proses menghapus kata-kata kunci yang tidak tepat untuk digunakan, seperti konjungsi, kata depan dan kata ganti, seperti kata yang, di, ke, dan,bisa, dll [6]. Pada tahap ini akan menghasilkan daftar *stop word.* Seluruh dari hasil *stop word* akan kembali disimpan pada data daftar kata sebagai daftar data penting[5].

3. Stemming

Stemming adalah proses pemisahan imbuhan yang terdapat pada sebuah kata untuk menghasilkan kata dasar. Biasanya dalam suatu kata untuk menggambarkan maksud yang lebih jelas akan menambahkan imbuhan terhadap kata tersebut, seperti contoh pada kata “mencari” menjadi “cari”. Keluaran dari proses ini adalah daftar kata dasar[5].

2.3 Pemilihan Kalimat

Dalam menentukan pemilihan kalimat, dalam penelitian ini menggunakan DAG (*Directed Acyclic Graph*) yaitu sebuah grafik yang terarah. Dijelaskan pada penelitian sebelumnya [7] cara merepresentasikan dokumen menggunakan DAG yaitu dokumen ini dibagi menjadi kalimat dan setiap kalimat dianggap sebagai simpul dari grafik ini. Bobot *edge* menunjukkan kesamaan kalimat. Grafik yang mewakili terdiri dari dua set, (V, E) di mana V adalah himpunan simpul dan E adalah himpunan *edge*. Untuk membuat grafik untuk dokumen sampel, dilakukan langkah-langkah berikut:

$$\forall_{si} \in Dokumen \;\; Add_{\,si}\; to\; V \ldots\ldots\ldots\ldots\ldots \quad (1)$$

$$\forall_{si,sj} \in V,\; jika_{\,si}\; adalah\;\; kalimat\; sebelum_{sj}\;\; di\; dalam\; dokumen, \ldots\ldots\ldots (2)$$

Maka, tatanan kronologis dalam pembentukan graf dari sebuah dokumen dapat menggunakan persamaan 1 dan 2, dimana $(_{si,sj})$ dapat membentuk himpunan E(edge).

1. Pembobotan tf-idf

Metode Term *Frequency-Inverse Document Frequency* (*TF-IDF*) adalah cara pemberian bobot untuk hubungan suatu kata terhadap dokumen keseluruhan. Perhitungan ini menggabungkan dua konsep untuk untuk mencari nilai bobotnya, yaitu *Term Frequency* (TF) merupakan frekuensi kemunculan kata pada suatu dokumen, sedangkan *document Frequency* adalah banyaknya kalimat dimana suatu kata muncul, perhitungan *Term Frequency* (TF) ditunjukkan pada persamaan 3. Bobot kata semakin besar jika kata tersebut sering muncul dalam suatu dokumen. [8].

Sistem pembobotan *tf-idf* klasik dalam penelitian [7] digunakan dengan beberapa asumsi yang serupa dengan model vektor klasik IR. Bobot *tf-isf* dihitung untuk setiap kalimat, di mana sj, j menunjukkan kalimat dan ki, i menunjukkan istilah *indeks*, rumus yang digunakan untuk menghitung bobot *tf-isf* ditunjukkan pada persamaan 4.

$$tf_{i,j} = \frac{freq_{i,j}}{max_i\, freq_{i,j}} \quad \ldots\ldots\ldots \quad (3)$$

$$isf_i = \log \frac{N}{n_i} \quad \text{.....(4)}$$

$tf_{i,j}$ dikatakan "frekuensi istilah" dari istilah *indeks* i dalam kalimat j, dan isf_i adalah "frekuensi kalimat terbalik" dari istilah *indeks* i, di mana N adalah jumlah semua kalimat dan n_i adalah jumlah kalimat yang mengandung k_i. Oleh karena itu bobot akan dihitung dengan mengalikan hasil perhitungan *tf* dan *isf* yang ditunjukkan pada persamaan 5.

$$w_{i,j} = tf_{i,j} \text{ x } isf_i \quad \text{.....} \quad (5)$$

Untuk membuat vektor dari setiap kalimat dan judul, judul akan dijadikan sebagai *query*. Bobot dalam membandingkan kalimat dengan judul dapat diperoleh dari persamaan 6.

$$w_{i,q} = (0.5 + \frac{0.5\ x\ freq_{i,q}}{max_i freq_{i,q}}) \text{ x } isf_i \quad \text{.....} \quad (6)$$

di mana $freq_{i,q}$, q adalah frekuensi baku dari istilah k_i dalam teks permintaan informasi q.

2. Perhitungan Cosinus Similarity

Model ruang vektor dan pembobotan *tf-idf* digunakan untuk merepresentasikan nilai dalam bentuk numerik dari sebuah dokumen, sehingga dapat dihitung kemiripan antar kalimat dalam sebuah dokumen. Kemiripan antar kalimat dihitung menggunakan perhitungan *similarity* pada persamaan 9, semakin besar hasil dari perhitungan *similarity,* maka kedua kalimat tersebut dinyatakan relevan atau memiliki kemiripan antar kalimatnya[9]. Pada persamaan 7 dan 8 menunjukkan rumus yang digunakan untuk mencari kemiripan pada setiap kalimat dengan judul [7]:

$$sim\ (s_j, q) = \frac{\overrightarrow{d_j}.\vec{q}}{|\overrightarrow{d_j}|\ x\ |\vec{q}|} \quad \text{.....} \quad (7)$$

$$sim\ (s_j, q) = \frac{\sum_{i=1}^{t} w_{i,j}\ x\ w_{i,q}}{\sqrt{\sum_{i=1}^{t} w^2{}_{i,j}}\ x\ \sqrt{\sum_{i=1}^{t} w^2{}_{i,q}}} \text{.....} \quad (8)$$

Demikian pula dengan kesamaan dua kalimat dapat dihitung menggunakan rumus:

$$sim\ (s_m, s_n) = \frac{\sum_{i=1}^{t} w_{i,m}\ x\ w_{i,n}}{\sqrt{\sum_{i=1}^{t} w^2{}_{i,m}}\ x\ \sqrt{\sum_{i=1}^{t} w^2{}_{i,n}}} \text{.....} \quad (9)$$

3. Perhitungan Topic Relation Factor

Ekstrak yang baik berisi kalimat yang mirip dengan judul teks.Kesamaan tersebut dapat dihiting menggunakan Faktor Relasi Topik (TRF) yaitu mempertimbangkan kesamaan rata-rata kalimat dalam ringkasan, dibagi dengan rata-rata maksimum. Misalkan, persamaan 10 menunjukkan perhitungan TR yaitu untuk menghitung kesamaan rata-rata dengan judul dalam suatu ringkasan,

$$TR_s = \frac{\sum_{s_j \in summary} sim(s_j, q)}{S} \text{.....} \quad (10)$$

Dengan menggunakan TR, untuk menghitung TRF dapat menggunakan rumus:

$$TRF_s = \frac{TR}{max_{\forall\ summary}(TR)} \text{.....} \quad (11)$$

TRF menunjukkan kesamaan dari ringkasan yang dibuat dengan judul dokumen. Dalam ringkasan di mana kalimat terkait erat dengan judul, nilai TRF dekat dengan 1. Namun jika kalimat terkait jauh dengan judul,maka TRF cenderung nol ini ditunjukkan pada persamaan 11.

4. Perhitungan Cohesion Factor

Pembobotan menggunakan *Cohesion Factor* adalah untuk menentukan apakah kalimat dalam sebuah ringkasan berisikan tentang informasi yang sama atau tidak ditunjukkan pada persamaan 12.

$$C_s = \frac{\sum \forall\, s_i, s_j \in summary\ subgraph\ w(s_i, s_j)}{N_s} \text{ ..... (12)}$$

di mana N_s adalah jumlah total *edge* dalam ringkasan *subgraph* ditunjukkan pada persamaan 13. N_s dapat dengan mudah dihitung. Misalkan simpul ringkasan adalah s_{s1}, s_{s2},. . . , s_{ss}, dan S adalah jumlah total kalimat dalam ringkasan. Kemudian N_s adalah jumlah *edge* dari s_{s1} ke s_{sj}, 1 < j ≤ S, ditambah jumlah *edge* dari s_{s2} ke s_{sj}, 2 <j ≤ S,. . .
jadi,

$$N_s = (S-1) + (S-2) + \cdots = \frac{(s)x(s-1)}{2} \text{ ..... (13)}$$

CF harus menunjukkan bagaimana ringkasan kalimat telah selesai dengan menggunakan persamaan 14.

$$CF_s = \frac{\log\,(C\ x\ 9{+}1)}{\log(M\ x\ 9{+}\ 1)} \text{ ..... (14)}$$

M adalah bobot maksimum dalam grafik,misalnya M adalah kesamaan maksimum kalimat.

5. Perhitungan Readibility Factor

Pembobotan ini digunakan untuk memudahkan dalam pembacaan pada sebuah ringkasan ysng ditunjukksn psds persamaan 15 dan 16, pada pembobotan ini akan membentuk rantai kalimat yang halus.
Misalkan *readability* ringkasan s dengan panjang S, katakanlah Rs, adalah:

$$R_s = \sum_{0 \le i \le S} W(s_i, s_i + 1) \text{ ..... (15)}$$

Oleh karena itu, faktor *readability* ringkasan s, dihitung seperti ini:

$$RF_s = \frac{R_s}{max_{\forall i}\, R_i} \text{ ..... (16)}$$

Perlu dicatat bahwa kami telah membuat asumsi bahwa panjang ringkasan adalah tetap. Dan maksimum dihitung di antara semua ringkasan yang mungkin dari panjang ringkasan itu.

Menemukan maksimum ini dapat dilakukan dalam waktu polinomial. Misalkan panjang ringkasan adalah S, sehingga tujuan menemukan ringkasan yang paling mudah dibaca adalah sama dengan menemukan jalur panjang S dengan bobot maksimum dalam grafik dokumen.

Saatnya untuk menemukan jalur terpanjang dengan jumlah node yang pasti dari DAG, yang ditunjukkan pada persamaan 17. Di sini kami bermaksud menemukan jalur maksimum dengan panjang tetap. Sederhana jika kita menganggap ringkasan sebagai s_{s1}, s_{s2},. . . , s_{ss},, Tujuannya adalah untuk memaksimalkan

$$sim_{s_{s1}, s_{s2}} + sim_{s_{s2}, s_{s3}} + \cdots + sim_{s_{s_{S-1}}, s_{s_S}} \text{ ..... (17)}$$

2.4 Algorima Genetika

Algoritma genetika merupakan suatu metode *heuristik* yang dikembangkan berdasarkan prinsip genetika dan proses seleksi alamiah Teori Evolusi Darwin. Metode optimasi dikembangkan oleh John Holland sekitar tahun 1960-an dan dipopulerkan oleh seorang mahasiswanya yaitu David Goldberg pada tahun 1980-an. Proses pencarian dalam algoritma ini sama seperti terpilihnya suatu individu yang mampu bertahan hidup dalam proses evolusi[10].Algoritma genetika merupakan teknik *search stochastic* yang berdasarkan mekanisme seleksi alam dan genetika natural. Yang membedakan algoritma ini dengan algoritma yang lain adalah pada algoritma genetika, penyelesaian dilakukan mulai dengan himpunan yang acak diawal proses yang biasanya disebut populasi[11]. Implementasi algoritma genetik menghasilkan banyak iterasi, jadi harus menggunakan kriteria berhentinya iterasi, yaitu ketika nilai *fitness* mencapai maksimum dalam 100 iterasi. Tahapan yang dilakukan pada Algoritma Genetika antara lain :

1. Inisialisasi populasi

Populasi awal dalam algoritma genetika dibentuk secara acak, sedangkan populasi berikutnya akan dibentuk oleh hasil dari operator algoritma genetika selama beberapa generasi[10]. Masing – masing individu yang terdapat dalam populasi disebut kromosom. Sebuah kromosom terdiri dari sebuah *string* berupa sederat bilangan binner 0

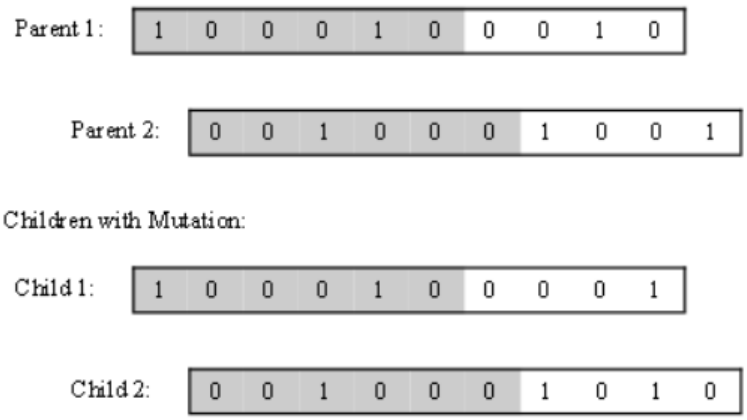
dan 1. Sebuah kromosom berkembang biak melalui berbagai iterasi yang berulang atau bisa juga disebut sebagai generasi[11].

2. Fungsi Fitness

Pada setiap generasi, kromosom akan mengalami proses evaluasi dengan menggunakan berhitungan dalam fungsi *fitness* (kebugaran). Rumus untuk menghitung suatu nilai *fitness* nya tergantung dengan masalah yang akan dioptimasi. Nilai *fitness* dari suatu kromosom akan menunjukkan kualitas kromosom dalam suatu populasi. Semakin besar nilai *fitness* suatu kromosom maka semakin besar pula kemungkinannya untuk dipertahankan ke dalam populasi berikutnya[10].

3. Crossover

Kromosom yang dibentuk dari generasi sebelumnya disebut sebagai anak (*offspring*). Demikian juga dengan pasangan kromosom pada generasi sebelumnya disebut sebagai induk (*parents*). Proses pembentukan generasi baru dapat dilakukan dengan cara melakukan proses penyilangan (*crossover*) yang ditunjukkan pada Gambar 2 [10]. Cara yang paling sederhana untuk melakukan *crossover* adalah memilih suatu *cut-point* dan menghasilkan *offsprin* dengan pengkombinasian segmen dari satu parent yeng berada di sebelah kiri dengan segmen dari parent satunya lagi yang sebelah kanan[11]. Generasi baru akan mewarisi sifat dari kedua induknya[10].



Gambar 2 ilustrasi penyilangan kromosom

4. Mutasi

Dalam proses alamiah, terkadang suatu makhluk hidup dapat mengalami proses mutasi yang dapat merubah sifat sifat aslinya, jadi berbagai kromosom akan mengalami perubahan spontan secara acak.[10] Cara yang paling sederhana dalam melakukan mutasi adalah dengan menukar satu atau lebih gen gen dalam kromosom.

Dalam algoritma genetika mutasi memiliki peran : a) menempatkan kembali gen gen yang hilang dari populasi sepanjang proses seleksi, agar mereka dapat dilibatkan kembali pada konteks yang berikutnya, b)menyediakan gen gen yang tidak terdapat pada *initial population* [11].

2.5 Skenario pengujian

Pengujian implementasi sistem akan dilakukan menggunakan persamaan *cosine similarity*. Pada penelitian [12] yang berjudul "*Automatic Text Summarization menggunakan Metode Graph dan Ant Colony Optimazation*"ditulis oleh I Wayan Adi Setyadi, dkk. Pengujian dilakukan dengan menggunakan 10 dokumen berita untuk diringkas melalui sistem. Pengujian hasil ringkasan dilakukan dengan cara menghitung nilai *cosine similarity* dari rangkuman manual yang dilakukan oleh peneliti dan rangkuman yang dihasilkan oleh sistem. Dengan menggunakan *cosine similarity* dapat mengukur kemiripan antara kedua rangkuman tersebut. Nilai dari kemiripan antara kedua rangkuman akan dijadikan sebagai tolak ukur keberhasilan implementasi sistem yang dilakukan.

3. Hasil Penelitian dan Pembahasan

Pada bab ini akan dilakukan implementasi terhadap aplikasi yang sudah dianalisa dan dirancang sebelumnya untuk mendapatkan sebuah hasil yang sesuai dengan tujuan penelitian skripsi ini, sehingga nantinya dengan berjalannya aplikasi ini dapat ditarik sebuah kesimpulan tentang bagaimana sistem peringkasan text secara otomatis atau biasa disebut dengan *Automatic Text Summarization System* ini berjalan dengan baik menggunakan metode *Genetic Algorithm.*

Sistem peringkasan berita secara otomatis atau *Automatic Text Summarization System* ini dapat membantu memudahkan pembaca untuk mendapatkan informasi yang disajikan dalam kalimat yag lebih ringkas dan mencangkup isi dari artikel berita tersebut.

Dalam implementasi desain, peneliti menggunakan template dari *templatemagz* dan penggunaan *Flask Webserver* pada *python*. Pada tampilan utama terdapat form untuk melakukan peringkasan teks berita dari pengguna. Input yang digunakan berupa *raw text* dari pengguna yang dapat diperoleh dari hasil *copy paste* suatu berita di portal berita *Online* atau dengan mengisi link URL berita pada kolom URL. Bagian yang bertanggung jawab menerima proses *query* dari *Input raw text* adalah *python* yang secara langsung berinteraksi dengan *framework web Flask*, juga berperan sebagai *mini webserver.* Pada *view Flask* sebelumnya juga menerima hasil pengolahan data pada *backend server* yang mana pada bagian tersebut terdapat fungsi untuk mentranslasikan antara *input raw text* ke *text* ataupun *URL* ke *text.*

Pada saat melakukan peringkasan teks, pengguna tidak akan mendapat *view* dari *controller*, untuk melihat proses ini sudah selesai apa belum dapat di cek di *mode debug* dari *web browser*. Pada bagian ini *Genetic Algorithm* bekerja pada bagian *model* yang disediakan oleh *python*. Hasil ringkasan akan ditampilkan pada halaman yang sama yaitu dibawah form saat pengguna mengisikan berita yang akan diringkas.

Setelah diimplementasikan berupa aplikasi berbasis *web* maka perlu dilakukan pengujian guna mengetahui sejauh mana kualitas dari hasil ringkasan yang dilakukan. Metode pengujian yang dilakukan peneliti merujuk pada jurnal [12] yang berjudul "*Automatic Text Summarization menggunakan Metode Graph dan Ant Colony Optimazation*"ditulis oleh I Wayan Adi Setyadi, dkk. Pengujian dilakukan dengan menggunakan 10 dokumen berita untuk diringkas melalui sistem. Pengujian hasil ringkasan dilakukan dengan cara menghitung nilai *cosine similarity* dari rangkuman manual yang dilakukan oleh peneliti dan rangkuman yang dihasilkan oleh sistem. Dengan menggunakan *cosine similarity* dapat mengukur kemiripan antara kedua rangkuman tersebut. Nilai dari kemiripan antara kedua rangkuman akan dijadikan sebagai tolak ukur keberhasilan implementasi sistem yang dilakukan.

Tabel 1. *Cosine Similarity* antara rangkuman manual dan Sistem

Rangkuman Nomor	Nilai *Cosine Similarity*	Prosentase
1	0.701	70,1 %
2	0.607	60,7 %
3	0.644	64,4 %
4	0.856	85,6 %
5	0.649	64,9 %
6	0.892	89,2 %
7	0.714	71,4 %
8	0.667	66,7 %
9	0.752	75,2 %
10	0.640	64 %
Rata – rata prosentase nilai		71,21 %

Pada tabel 1 merupakan hasil yang diperoleh setelah membandingkan antara ringkasan yang dilakukan oleh sistem dengan ringkasan yang dilakukan secara manual mendapatkan nilai rata rata keseluruhan sebesar 71,21%. Dengan pencapaian prosentase tersebut dapat disimpulkan bahwa ringkasan yang dilakukan oleh sistem sebagian besar sama dengan ringkasan yang dilakukan secara manual. Hal ini menunjukkan bahwa sistem yang dibuat oleh peneliti layak untuk digunakan oleh pengguna.

4. Kesimpulan

Penggunaan sistem *Automatic Summarizatio*n membantu pembaca untuk menemukan inti dari sebuah artikel berita. Sistem ini bekerja dengan menggunakan algoritma genetika untuk pencarian kalimat yang merupakan inti dari suatu dokumen. Dengan adanya sistem ini, pengguna dapat menghemat waktu untuk mendapatkan informasi yang diinginkan. Dari hasil pengujian yang dilakukan dengan cara membandingkan ringkasan sistem dengan ringkasan manual yang dilakukan oleh peneliti menggunakan perhitungan *cosine similarity*,memberikan hasil yang tinggi yaitu rata – rata nilainya sebesar 71,21%. Sehingga dapat ditarik kesimpulan bahwa hasil ringkasan oleh sistem memiliki kemiripan pada hasil ringkasan yang dilakukan secara manual. Dengan hasil pengujian yang cukup tinggi, menandakan bahwa sistem yang dibangun oleh peneliti dapat digunakan untuk pengguna. Berikut adalah pencapaian yang diperoleh dalam pembangunan sistem ini :

1. Tercapainya implementasi peringkasan dokumen berita menggunakan algoritma genetika dengan tingkat kemiripan yang tinggi jika diukur mdengan ringkasan yang dilakukan yaitu dengan nilai prosentase 71,21%.
2. Tercapainya proses peringkasan dokumen yang singkat,efisien,dan sesuai dengan isi berita.

Refrensi

[1] P. M. Berita, “Ekstraksi trending issue dengan pendekatan distribusi kata pada pembobotan term untuk peringkasan multi-dokumen berita,” vol. 14, pp. 180–189, 2016.
[2] R. Azhar *et al.*, “PEMBOBOTAN KATA BERDASARKAN KLASTER PADA OPTIMISASI COVERAGE , DIVERSITY DAN COHERENCE,” vol. II, no. 3, pp. 170–178, 2016.
[3] B. Indonesia, A. Genetika, and D. A. N. A. Genetika, “Implementasi Automated Text Summarization untuk Dokumen Tunggal IMPLEMENTASI AUTOMATED TEXT SUMMARIZATION UNTUK DOKUMEN TUNGGAL BERBAHASA INDONESIA DENGAN MENGGUNAKAN GRAPH-BASED,” no. July, 2014.
[4] D. A. Prabowo, M. Fhadli, M. A. Najib, and H. A. Fauzi, “TF-IDF- ENHANCED GENETIC ALGORITHM UNTUK EXTRACTIVE AUTOMATIC TEXT SUMMARIZATION,” vol. 3, no. 3, pp. 208–215, 2016.
[5] P. M. Prihatini, “PERANCANGAN SUBSISTEM PENGOLAHAN PERTANYAAN UNTUK,” vol. 16, no. 1, pp. 53–57, 2016.
[6] S. Program and P. K. Malang, “Improving Multi-Document Summary Method Based on Sentence Distribution,” vol. 14, no. 1, pp. 286–293, 2016.
[7] V. Qazvinian and L. S. Hassanabadi, “Summarizing Text with a Genetic Algorithm-Based Sentence Extraction,” pp. 1–10, 1997.
[8] “INDONESIAN_TEXT_DOCUMENT_SUMMARIZATION_M.” .
[9] P. K. Praktek, T. Akhir, and D. A. N. Skripsi, “PENGUNAAN METODE COSINESIMILARITY PADA SISTEM.”
[10] Z. Zukhri, *Algoritma Genetika*, 1st ed. Yogyakarta: Andi Offset, 2014.
[11] Fadlisyah, Arnawan, and Faisal, *Algoritma Genetika*, 1st ed. Yogyakarta: Graha Ilmu, 2009.
[12] I. W. A. Setyadi, D. C. Khrisne, and I. M. A. Suyadnya, “Automatic Text Summarization Menggunakan Metode Graph dan Ant Colony Optimization,” vol. 17, no. 1, pp. 124–130, 2018.