REPOSITOR, Vol. 5, No. 2, Mei 2023, Pp. 709-716
ISSN : 2714-7975
E-ISSN : 2716-1382

Improvisasi Algoritma Dijkstra Pada Peringkasan Teks Otomatis Untuk Artikel Politik

Giffari Zakawaly*[1], Nur Hayatin[2], Vinna Rahmayanti Setyaning Nastiti[3]
[1,2,3]Universitas Muhammadiyah Malang
giffarizz@gmail.com*[1], noorhayatin@gmail.com[2], vinastiti@umm.ac.id[3]

Abstrak

Tidak jarang kita menemukan suatu informasi atau artikel yang terlalu panjang. Dan untuk mengatasi masalah tersebut, salah satu cara yang bisa digunakan yaitu peringkasan teks. Metode peringkasan teks yang yang saat ini banyak dan biasa digunakan bisa dikelompokkan ke dalam tiga kategori, dan salah satunya yaitu metode berbasis graf. Dan algoritma Dijkstra memiliki tahapan yang paling sederhana. Improvisasi dilakukan dengan cara menambahkan 3 fitur pembobotan lain yaitu word freuency, position, dan resemblance to the title. Improvisasi yang dilakukan tidak terlalu berpengaruh terhadap ringkasan yang dihasilkan oleh sistem. Hal ini ditunjukkan dengan nilai evaluasi ROUGE-1 tanpa improvisasi 0.48487 dan dengan improvisasi 0.47513.

Kata Kunci: *Peringkasan Teks, Dijkstra, Improvisasi, Fitur Pembobotan*

Abstract

It's often for us to find information or articles that are too long. And to solve this problem, one way that can be used is text summarization. The text summarization methods that are currently widely used can be grouped into three categories, one of which is the graph-based method. And Dijkstra algorithm has the simplest stages. Improvisation is done by adding three other weighting features, namely word frequency, position, and resemblance to the title. The improvisation that's done doesn't really affect the summary produced by the system. Shown by the ROUGE-1 evaluation value without improvisation is 0.48487 and with improvisation is 0.47513.

Keywords: *Text Summarization, Dijkstra, Improvisation, Weighting Features*

1. Pendahuluan

Pada era seperti sekarang ini kita sudah sangat dimudahkan untuk mencari informasi apapun yang kita mau. Dengan adanya internet kita bisa dengan cepat mengetahui sebuah informasi mengenai sesuatu yang baru saja terjadi. Baik itu melalui sosial media yang kita miliki, atau melalui berbagai macam situs berita yang bisa kita akses dimana dan kapan saja yang kita mau, selama kita memiliki koneksi internet. Meskipun demikian, tidak jarang kita menemukan suatu informasi atau artikel yang terlalu panjang, sehingga memerlukan waktu yang cukup lama untuk membaca keseluruhan informasi yang kita dapat. Dan untuk mengatasi masalah tersebut, salah satu cara yang bisa digunakan yaitu peringkasan teks.

Metode peringkasan teks yang yang saat ini banyak digunakan bisa dikelompokkan ke dalam tiga kategori, salah satunya yaitu metode berbasis graf. Penelitian peringkasan teks berbasis graf sudah pernah dilakukan sebelumnya dengan berbagai metode yang berbeda, beberapa diantaranya seperti algoritma Dijkstra [1], algoritma genetika [2], serta algoritma *Pagerank* dan *Sparse Non Negative Matrix Factorization* (SNMF) [3].

1.1 Peringkasan Teks

Peringkasan teks [4] merupakan proses penulisan ataupun menyatakan ulang suatu teks tertentu menjadi lebih pendek dengan cara memahaminya terlebih dahulu. Seperti yang juga dijelaskan pada penelitian [5][6], ringkasan diartikan sebagai sebuah teks yang terdiri dari satu atau lebih kalimat yang berisi informasi penting dari suatu artikel atau dokumen asli. Panjang dari ringkasan yang dihasilkan tidak lebih dari setengah panjang dokumen asli atau bahkan bisa lebih pendek lagi.

Ringkasan yang dihasilkan oleh manusia biasanya menggambarkan seluruh informasi-informasi penting dari dokumen aslinya secara akurat. Usaha dan waktu yang diperlukan juga bervariasi tergantung pada pemahaman dan tingkat kecerdasan masing-masing individu untuk menghasilkan ringkasan dari teks. Pasti akan sangat membosankan atau bahkan hampir mustahil bagi manusia untuk merangkum teks dalam jumlah yang sangat banyak secara manual dan individu. Meskipun ringkasan yang dibuat manual oleh manusia memiliki tingkat akurasi yang lebih tinggi, tetapi kita perlu melakukan peringkasan teks secara otomatis untuk menyelesaikan masalah seperti yang dijelaskan di atas.

1.2 Berita

Berasal dari bahasa sansekerta *vrit*, atau dalam bahasa Inggris *write*, berita memiliki arti sebenarnya ada atau terjadi. Sebagian juga menyebut dengan *writta*, yang berarti kejadian atau yang telah terjadi. Dalam Kamus Besar Bahasa Indonesia, berita diperjelas menjadi laporan mengenai kejadian atau peristiwa yang hangat [7][8].

Pada penelitian [7] juga dijelaskan bahwa dalam sebuah berita terdapat dua aspek substansial, yaitu nilai faktualitas dan imparsialitas. Faktualitas itu sendiri diartikan bagaimana kualitas informasi sebuah berita yang disajikan. Faktualitas terkait pada tiga hal, antara lain kebenaran (*truth*), relevansi (*relevance*) serta *informativeness*. Sedangkan imparsialitas, meninjau apakah suatu berita memiliki keberpihakan pada satu pihak atau tidak. Imparsialitas itu sendiri dapat dilihat dalam dua hal, yaitu netralitas dan keberimbangan.

Seiring dengan berkembangnya teknologi, kebiasaan untuk mendapatkan informasi berupa berita mulai bergeser dari media cetak ke media *online*. Selain karena lebih mudah untuk diakses, kebiasaan ini juga terjadi karena sesuai dengan gaya hidup masyarakat milenial saat ini. Tidak bisa dipungkiri bahwa sekarang kebanyakan informasi yang kita didapatkan berasal dari portal-portal berita *online* yang sudah mulai banyak bermunculan, atau bahkan berasal dari sosial media yang kita gunakan.

Pemilihan berita politik pada penelitian ini dilandasi dengan peristiwa politik yang selalu menarik perhatian media massa sebagai bahan utama liputan. Hal ini dapat terjadi karena dua faktor yang saling berkaitan [7]. Pertama, dewasa ini politik berada di era mediasi yakni media massa, sehingga hampir mustahil kehidupan politik dipisahkan dari media massa. Kedua, peristiwa politik dalam bentuk tingkah laku dan pernyataan aktor politik lazimnya selalu mempunyai nilai berita.

1.3 Peringkasan Teks Otomatis

Peringkasan otomatis sebenarnya berarti memilih secara otomatis bagian-bagian penting yang bisa berbentuk paragraf ataupun kalimat dari satu atau banyak teks lengkap [4][9]. Tidak mudah untuk mengimplementasikan program aplikasi yang bisa merangkum teks secara otomatis seperti halnya manusia. Peringkasan teks otomatis harus didefinisikan secara berbeda dari peringkasan teks manual yang dijelaskan sebelumnya. Khususnya bagaimana proses atau metode memilih bagian-bagian penting dari suatu teks lengkap atau dokumen.

Metode peringkasan teks yang yang saat ini banyak digunakan bisa dikelompokkan ke dalam tiga kategori, salah satunya yaitu metode berbasis graf. Penelitian peringkasan teks berbasis graf sudah pernah dilakukan sebelumnya dengan berbagai metode yang berbeda, beberapa diantaranya seperti algoritma Dijkstra [1], algoritma genetika [2], serta algoritma *Pagerank* dan *Sparse Non Negative Matrix Factorization* (SNMF) [3]. Dokumen yang digunakan pada ketiga penelitian tersebut akan direpesentasikan dalam bentuk graf berbobot, yang kemudian akan dihitung bobot relasi setiap antar kalimat. Selanjutnya pada penelitian [3] proses akan dilanjutkan dengan melakukan *ranking* pada kalimat, kemudian digunakan metode SNMF untuk mendapatkan kalimat inti dari masing-masing cluster yang dihasilkan untuk kemudian disusun menjadi ringkasan. Proses lanjutan pada penelitian [2] juga terdiri dari beberapa tahap yaitu pengkodean, seleksi, rekombinasi dan mutasi dimana akan dilakukan penduplikasian dan pertukaran bagian-bagian dari string. Sedangkan pada penelitian [1] proses akan dilanjutkan dengan algoritma Dijkstra yang akan langsung menghasilkan ringkasan.

Setelah mengetahui penelitian-penelitian sebelumnya, terlihat bahwa algoritma Dijkstra memiliki tahapan yang lebih sederhana dibandingkan metode pada penelitian [2] dan [3]. Meskipun sederhana, ringkasan yang dihasilkan tidak jauh berbeda. Hasil dari penelitian [1] menunjukkan performa metode yang digunakan memiliki nilai *F-Measure* 0.331726. Hal ini bisa dikarenakan proses pembobotan yang dilakukan hanya menggunakan *Term Frequency Inverse*

Sentences Frequency (TF-ISF). Sedangkan pada penelitian [10] disebutkan dibutuhkan teknik pembobotan kalimat yang handal untuk dapat menghasilkan ringkasan berita yang baik.

Oleh karena itu dalam penelitian ini akan dilakukan improvisasi dengan menambahkan beberapa fitur lain yang akan dipertimbangkan dalam proses pembobotan. Pada penelitian [5] dan [10] disebutkan ada empat fitur pembobotan yang sangat baik untuk dikombinasikan pada dokumen dengan karakter teks pendek dan terstruktur, yaitu *word frequency*, TF-IDF, posisi kalimat, dan *Resemblance to The Title*.

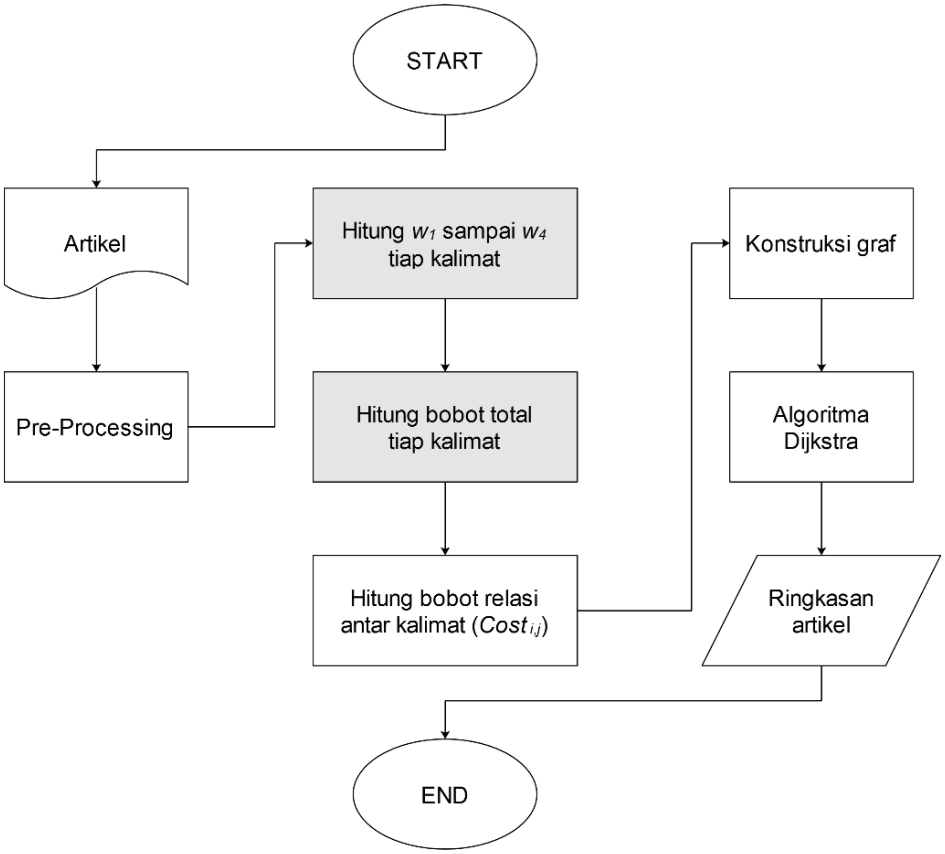
Diharapkan improvisasi yang akan dilakukan bisa meningkatkan performa dari ringkasan yang dihasilkan. Jika pada penelitiaan sebelumnya teks yang diringkas adalah artikel ilmiah berbahasa Inggris, maka pada penelitian ini adalah artikel politik berbahasa Indonesia.

1.4 Algoritma Dijkstra

Algoritma Dijkstra biasa digunakan untuk mencari jalur terpendek dalam urutan peningkatan panjang jalur yang dilalui. Ini menjadi sebuah solusi untuk mengetahui jalur terpendek dari node sumber atau awal ke node tujuan dalam peta dari satu kumpulan node. Algoritma akan menentukan jalur terpendek di antara jalur dari node sumber ke node yang berdekatan terlebih dahulu. Node terminal dari jalur kemudian dianggap sebagai node perantara [11]. Algoritma kemudian mencari jalur terpendek berikutnya dari node perantara ke node yang berdekatan. Iterasi berlanjut sampai setiap node dilalui. Ciri dari algoritma Dijkstra adalah setiap sub-path dari shortest path merupakan jalur terpendek dari starting node ke terminal node, sehingga dapat diperoleh jalur terpendek dari source node ke goal node manapun pada path tersebut.

2. Metode Penelitian

Adapun metodologi yang digunakan untuk melakukan peringkasan teks otomatis dalam penelitian ini dapat dilihat seperti pada Gambar 1.



Gambar 1. Metodologi Penelitian

2.1 Data

Data yang digunakan dalam pembuatan tugas akhir ini adalah data *plaintext* berupa *file .txt* yang berisi artikel politik yang dikumpulkan dari beberapa portal berita *online*. Dengan artikel yang akan digunakan sebanyak 100 artikel politik.

2.2 Pre-Processing

Text mining yang efektif bergantung pada metode p*re-processing* yang digunakan [12]. *Pre-processing* merupakan suatu proses untuk merubah data teks menjadi sebuah *term* indeks yang akan digunakan dalam proses pembobotan [13][14]. Tahap ini juga bisa diartikan sebagai sebuah tahapan dimana dokumen mentah akan diolah sehingga menjadi dokumen yang siap diproses untuk langkah selanjutnya [15].

Dan dalam *pre-processing* pada umumnya terdapat beberapa tahapan yang akan dilakukan. Tahap yang pertama adalah *parsing*, yaitu suatu proses untuk menentukan data teks apa yang akan digunakan. Kemudian dilanjutkan dengan *tokenization* dimana dilakukan proses untuk memisahkan setiap kata yang ada pada kalimat dan merubah semua huruf kapital yang ada menjadi huruf kecil [13]. Selanjutnya yaitu *stopword removal* dan *stemming*. *Stopword removal* adalah proses menghapus kata-kata yang tidak penting atau tidak perlu digunakan sebagai penciri dokumen. Sedangkan *stemming* adalah proses penguraian berbagai bentuk kata baik itu prefix, sufix, maupun gabungan antara prefix dan sufix (confix), menjadi bentuk kata dasarnya [15].

2.3 Pembobotan Kalimat

Dalam pembuatan tugas akhir akan digunakan empat teknik pembobotan kalimat, yaitu *Term Frequency Inverse Sentences Frequency* (TF-ISF) seperti pada penelitian [1] dan [16], dan juga *Word Frequency* (WF), posisi kalimat, dan kemiripan kalimat terhadap judul berita (*Resemblance to the Title*) yang dilakukan pada penelitian [5] dan [10]. Kemudian nilai bobot dari keempat teknik tersebut akan dijumlahkan untuk mendapatkan bobot total dari tiap kalimat.

Pembobotan pertama (***w₁***) yaitu TF-ISF yang merupakan sebuah adaptasi dari TF-IDF sehingga memiliki konsep yang tidak jauh berbeda. Jika TF-IDF digunakan untuk menghitung bobot sebuah dokumen, maka TF-ISF digunakan untuk melakukan pembobotan pada kalimat [1], dengan Persamaan 1.

$$isf_t = log\left(\frac{N}{sf_{(t)}}\right) \quad (1)$$

Dimana *N* adalah banyaknya kalimat dalam dokumen dan ***sf(t)*** adalah jumlah kalimat dimana kata ***t*** muncul. Nilai ***isf*** digunakan dalam perhitungan nilai TF-ISF seperti pada Persamaan 2.

$$TF - ISF_{(t,s_i)} = tf_{(t,s_i)} \times isf_{(t)} \quad (2)$$

Dimana ***tf(t,si)*** adalah jumlah kata ***t*** yang muncul pada kalimat ***i***. Sehingga bobot kalimat bisa dihitung dengan Persamaan 3.

$$w_1(s_i) = \frac{\sum TF - ISF(s_i)}{|s_i|} \quad (3)$$

Pembobotan kedua (***w₂***) dihitung dari nilai kemiripan kalimat ***sᵢ*** terhadap *Word Frequency List* (*WFList*) menggunakan *cosine similarity*, dimana *WFList* didapatkan dari sejumlah *term* dengan nilai WF yang memenuhi nilai ambang (*threshold*). Kalimat dan *WFList* akan direpresentasikan sebagai *vector*. Idealnya, kedua *vector* yang akan dibandingkan memiliki panjang yang sama. Kemudian jika kedua *vector* memiliki panjang yang berbeda, maka diperlukan *global order*. Ada kemungkinan diamana kata-kata dalam *WFList* sudah ditambahkan pada *global order* karena jua muncul pada kalimat ***sᵢ*** seperti pada contoh berikut.

Kalimat s_i : Informatika salah satu prodi Fakultas Teknik UMM
WFList : UMM Informatika Malang Mahasiswa

Sebelum bisa menghitung nilai *cosine similarity*, perlu dilakukan penambahan elemen 0 pada *vector* yang lebih pendek, yang artinya kata tersebut tidak ada pada *vector*. Untuk lebih jelasnya ditunjukkan pada Tabel 1.

Tabel 1. Cosine Similarity dengan panjang vector berbeda

Global list	Vector s_i	Vector *WFList*
informatika	1	1
salah	1	0
satu	1	0
prodi	1	0
fakultas	1	0
teknik	1	0
umm	1	1
malang	0	1
mahasiswa	0	1

Setelah mendapatkan panjang *vector* yang sama maka proses perhitungan *cosine similarity* bisa dilakukan menggunakan Persamaan 4.

$$w_2(s_i) = Sim(s_i, WFList) \tag{4}$$

Pembobotan ketiga (***w₃***) menggunakan fitur posisi kalimat. Dimana kalimat yang berada diawal dokumen mendapat bobot lebih besar dibandingkan dengan kalimat yang berada diakhir dokumen. Pembobotan ketiga bisa dilakukan dengan menggunakan Persamaan 5 berikut.

$$w_3(s_i) = \frac{1}{\sqrt{POS(s_i)}} \tag{5}$$

Pembobotan keempat (***w₄***) didapatkan dengan membagi antara jumlah *term* judul yang muncul pada kalimat (NTW) dengan jumlah seluruh term yang ada pada judul (T) [5][10]. Pembobotan keempat bisa dilakukan menggunakan Persamaan 6 sebagai berikut.

$$w_4(s_i) = \frac{NTW}{T} \tag{6}$$

Setelah mendapatkan keempat nilai bobot kemudian akan dihitung total bobot yang dimiliki oleh kalimat ***i*** dengan menggunakan Persamaan 7 berikut.

$$weight(s_i) = w_1(s_i) + w_2(s_i) + w_3(s_i) + w_4(s_i) \tag{7}$$

2.4 Algoritma Dijkstra

Sebelum algoritma digunakan, dokumen terlebih dahulu akan direpresentasikan dalam bentuk graf. Pada umumnya graf bisa digambarkan sebagai kumpulan pasangan *(V,E)*, dimana *V* merupakan kumpulan dari *vertex* dan *E* merupakan kumpulan dari *edge* yang menghubungkan *vertex* [17]. Dimana *V* akan merepresentasikan setiap kalimat pada dokumen dan *E* merepresentasikan relasi atau keterkaitan antar kalimat. Bobot dari *E* bisa didapatkan dengan menggunakan Persamaan 8 berikut.

$$Cost_{i,j} = \frac{(i-j)^2}{overlap_{i,j} \times weight_j} \tag{8}$$

Dimana nilai $overlap_{(i,j)}$ merupakan jumlah kata yang sama antara kalimat ke-*i* dan kalimat ke-*j*. Setelah mendapatkan model graf, algoritma Dijkstra akan digunakan untuk menemukan jalur

terpendek dari setiap kalimat pada dokumen. Ringkasan yang dihasilkan merupakan semua kalimat yang dilewati oleh jalur terpendek.

3. Hasil Penelitian dan Pembahasan

Pengujian akan dilakukan dengan cara mengukur performa hasil ringkasan metode awal dan hasil ringkasan metode yang sudah diimprovisasi menggunakan metode ROUGE, yaitu metode evaluasi yang bisa menilai kualitas ringkasan sistem dengan membandingkannya terhadap ringkasan manusia atau pakar. Jenis-jenis metode pada ROUGE adalah ROUGE-N, ROUGE-L, ROUGE-W, ROUGE-S dan ROUGE-SU [18]. Evaluasi menggunakan model unigram dari ROUGE-N, yaitu ROUGE-1 berkolerasi baik terhadap hasil evaluasi manusia berdasarkan berbagai statistic [19]. Selain itu ROUGE-1 juga sangat baik digunakan untuk mengevaluasi ringkasan artikel pendek [20], oleh karena itu penelitian ini akan menggunakan evaluasi ROUGE-1 dengan teknik penghitungan berdasarkan Persamaan 9 berikut.

$$ROUGE - N = \frac{\sum_{S \in RefSum} \sum_{gram_n \in S} count_{match}(gram_n)}{\sum_{S \in RefSum} \sum_{gram_n \in S} count(gram_n)} \quad (9)$$

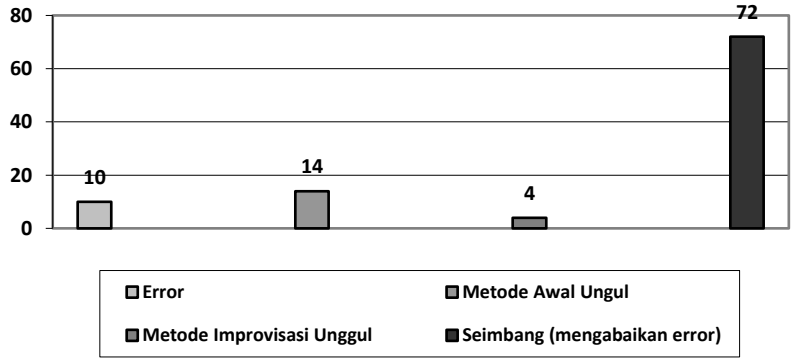
Ringkasan yang dihasilkan oleh sistem akan dibandingkan dengan ringkasan yang dibuat secara manual oleh pakar. Kemudian kedua nilai ROUGE-1 dari hasil ringkasan metode yang sudah diimprovisasi akan dibandingkan dengan hasil ringkasan metode awal.

3.1 Data

Adapun 2 *dataset* yang digunakan pada penelitian ini adalah100 *file .txt* yang berisi artikel politik dari berbagai portal berita online dengan panjang artikel yang bervariasi, dan 100 *file .txt* yang berisi ringkasan manual yang akan digunakan sebagai acuan untuk mengitung nilai ROUGE-1 dari tiap hasil ringkasan sistem.

3.2 Hasil

Setelah dilakukan penelitian dari 100 artikel yang ada, terdapat 10 artikel yang mengalami *fatal error: maximum execution time exceeded* ketika dilakukan proses peringkasan. Hal ini disebabkan karena artikel terlalu panjang dan sistem perlu melakukan banyak proses *looping* di setiap tahapannya. Untuk mengatasi masalah ini sudah dilakukan percobaan ulang dengan menambahkan *maximum execution time* menjadi 180 detik dan 300 detik. Namun *error* yang sama tetap terjadi sehingga nilai evaluasi dianggap 0. Hasil performa sistem dapat dilihat pada Gambar 2.



Gambar 2. Grafik Hasil Performa Sistem

Tanpa mengabaikan hasil *error*, sebanyak 82 artikel memiliki nilai evaluasi yang sama. Dimana artikel-artikel ini memiliki hasil ringkasan sistem yang sama persis, ataupun memiliki jumlah kalimat penyusun ringkasan yang sama namun berbeda pada kalimat yang dipilih. Justru sebanyak 14 hasil ringkasan yang menggunakan metode awal mendapatkan nilai evaluasi yang lebih tinggi dibandingkan dengan metode improvisasi, namun jumlah kalimat yang menjadi ringkasan lebih banyak dari metode improvisasi.

Dan sebaliknya, hasil ringkasan metode improvisasi hanya lebih unggul sebanyak 4 artikel saja dengan jumlah kalimat ringkasan yang juga lebih banyak. Perbandingan antar metode nilai

evaluasi ROUGE-1 tiap artikel dengan mengabaikan *error* dan nilai yang sama dapat dilihat pada Tabel 2. Dan perbandingan kedua metode secara umum seperti yang ditunjukkan Tabel 3.

Tabel 2 Perbandingan Nilai Evaluasi Tiap Artikel

Artikel ID	Improv	Non-improv	Gap	Artikel ID	Improv	Non-improv	Gap
1	0.37589	0.38298	-0.00709	61	0.31288	0.41718	-0.10430
27	0.05556	0.26984	-0.21428	64	0.50538	0.51613	-0.01075
38	0.40385	0.42949	-0.02564	66	0.44444	0.43210	0.01234
41	0.46000	0.53000	-0.07000	72	0.43443	0.41803	0.01640
47	0.23711	0.32990	-0.09279	74	0.38406	0.39130	-0.00724
48	0.23585	0.35849	-0.12264	75	0.41135	0.42553	-0.01418
49	0.13953	0.25581	-0.11628	89	0.61290	0.56774	0.04516
56	0.35417	0.39583	-0.04166	93	0.42857	0.37594	0.05263
60	0.37405	0.51145	-0.13740	98	0.45806	0.49677	-0.03871

Tabel 3 Perbandingan Metode Secara Umum

	Dengan *Error*		Mengabaikan *Error*	
	Improv	Non-Improv	Improv	Non-Improv
Minimal ROUGE-1	0	0	0.05556	0.26984
Maksimal ROUGE-1	0.88136	0.88136	0.88136	0.88136
Rata-rata ROUGE-1	0.42762	0.43638	0.47513	0.48487

Dari Tabel diatas bisa dilihat bahwa nilai evaluasi rata-rata ROUGE-1 dari metode awal, 0.43638, sedikit lebih tinggi dari nilai evaluasi rata-rata metode improvisasi, 0.42762. Jika mengabaikan 10 artikel yang mengalami *error*, maka nilai evaluasi rata-rata ROUGE-1 akan menjadi 0.48487 untuk metode awal dan 0.47513 untuk metode improvisasi. Namun keduanya memiliki nilai evaluasi tertinggi yang sama, yaitu 0.88136.

4. Kesimpulan

Dari hasil penelitian tentang improvisasi algoritma Dijkstra pada peringkasan teks otomatis untuk artikel politik dapat diambil kesimpulan sebagai berikut:

1. Setelah didapatkan hasil pengujian pembobotan kalimat, diketahui dari 100 artikel yang diuji hanya 10 artikel yang mengalami error dan tidak bisa melanjutkan ke tahapan algoritma Dijkstra. Sehingga bisa disimpulkan improvisasi terhadap metode awal tetap bisa dilakukan pada tahap pembobotan kalimat. Improvisasi pada proses pembobotan awal yang hanya menggunakan TF-ISF saja dilakukan dengan menambahkan 3 fitur pembobotan lain yaitu word freuency, position atau posisi kalimat, dan resemblance to the title atau kemiripan dengan judul.
2. Pada pengujian evaluasi ROUGE-1 diperoleh nilai evaluasi dari setiap artikel yang diuji seperti yang ditampilkan pada Tabel 2 sebelumnya. Dari hasil tersebut bisa disimpulkan improvisasi pembobotan yang dilakukan pada algoritma Dijkstra tidak terlalu berpengaruh terhadap ringkasan yang dihasilkan oleh sistem. Dimana nilai evaluasi rata-rata ROUGE-1 yang diperoleh dari metode awal sedikit lebih tinggi dibandingkan dengan metode improvisasi, yaitu 0.48487 berbanding 0.47513.

Berikut beberapa saran untuk penelitian-penelitian selanjutnya:

1. Menyederhanakan program pada saat *pre-processing* dan pembobotan agar tidak terjadi *fatal error: maximum execution time exceeded* ketika artikel yang diringkas terlalu panjang.
2. Tidak menggunakan kalimat pertama dan terakhir pada artikel sebagai titik awal dan tujuan. Gunakan cara pemilihan lain agar fitur-fitur pembobotan yang ditambahkan bisa berdampak lebih maksimal.

Referensi

[1] Y. B. Prasetyo, G. Virginia, and A. Rahmat, "Implementasi Algoritma Dijkstra untuk Peringkasan Otomatis Artikel Ilmiah Berbahasa Inggris," *INFORMATIKA*, vol. 11, no. 1, pp. 89–97, 2015.

[2] G. A. Pradnyana and I. K. A. Mogi, "Implementasi Automated Text Summarization untuk Dokumen Tunggal Berbahasa Indonesia dengan Menggunakan Graph-Based Summarization Algorithm dan Algoritma Genetika," *NERO*, vol. 1, no. 2, pp. 33–46, 2014.

[3] I. Darmawan, R. A. Harianto, and H. Armanto, "Peringkasan Teks Model Graf pada Single Dokumen dengan Metode Sparse Non Negative Matrix Factorization," in *SENTRA*, 2017, no. 5.

[4] T. Jo, *Text Summarization. Studies in Big Data*, 45th ed. Cham: Springer International Publishing, 2019.

[5] N. Hayatin, C. Fatichah, and D. Purwitasari, "Pembobotan Kalimat Berdasarkan Fitur Berita dan Trending Issue untuk Peringkasan Multi Dokumen Berita," *JUTI*, vol. 13, no. 1, pp. 38–44, 2015.

[6] J. S. Saputra, M. Fachrurrozi, and Yunita, "Peringkasan Teks Berita Berbahasa Indonesia Menggunakan Metode Latent Semantic Analysis (LSA) dan Teknik Steinberger & Jezek," in *Computer Science and ICT*, 2017, vol. 3, no. 1, pp. 215–219.

[7] A. C. Bahrun, "Analisis Isi Berita Politik Pilkada Gowa 2015 Pada Rubrik Citizen Reporter Portal Berita Online www.gosulsel.com (Suatu Studi Tentang Jurnalisme Warga Dari Perspektif Kelengkapan Berita)," *Kareba J. Ilmu Komun.*, vol. 8, no. 1, pp. 169–187, 2019.

[8] P. G. Somantri, A. Komarudin, and R. Ilyas, "Peringkasan Teks Otomatis Berita Berdasarkan Klasifikasi Kalimat Menggunakan Support Vector Machine," in *SNATIF*, 2018, pp. 57–62.

[9] N. Nazari and M. A. Mahdavi, "A survey on Automatic Text Summarization," *JADM*, vol. 7, no. 1, pp. 121–135, 2019.

[10] S. Verdianto, A. Z. Arifin, and D. Purwitasari, "Strategi pemilihan kalimat pada peringkasan multi dokumen," *Nusant. J. Comput. its Appl.*, vol. 2, no. 7, pp. 1–5, 2016.

[11] G. Qing, Z. Zheng, and X. Yue, "Path-planning of Automated Guided Vehicle based on Improved Dijkstra Algorithm," in *Chinese Control And Decision Conference (CCDC)*, 2017, pp. 7138–7143.

[12] D. Petrovic and M. Stankovic, "The Influence Of Text Preprocessing Methods And," *Facta Univ. Ser. Math. Informatics*, vol. 34, no. 5, pp. 973–994, 2019.

[13] D. A. Prabowo, M. Fhadli, M. A. Najib, and H. A. Fauzi, "Tf-Idf- Enhanced Genetic Algorithm Untuk Extractive Automatic Text Summarization," *JTIIK*, vol. 3, no. 3, pp. 208–215, 2016.

[14] D. Andriani and M. T. Furqon, "Peringkasan Teks Otomatis Pada Artikel Berita Hiburan Berbahasa Indonesia Menggunakan Metode BM25," *JPTIIK*, vol. 3, no. 3, pp. 2603–2610, 2019.

[15] B. Zaman and E. Winarko, "Analisis Fitur Kalimat untuk Peringkas Teks Otomatis pada Bahasa Indonesia," *IJCCS*, vol. 5, no. 2, pp. 60–68, 2011.

[16] I. Boban, A. Doko, and S. Gotovac, "Sentence retrieval using Stemming and Lemmatization with Different Length of the Queries," *Adv. Sci. Technol. Eng. Syst. J.*, vol. 5, no. 3, pp. 349–354, 2020.

[17] K. Ruohonen, *Graph theory*. 2013.

[18] A. N. Ammar, "Peringkasan Teks Ekstraktif Menggunakan Binary Firefly Algorithm," vol. 5, no. September, pp. 31–42, 2020.

[19] I. P. G. H. Saputra, "Peringkasan Teks Otomatis Untuk Dokumen Bahasa Bali Berbasis Metode Ektraktif," vol. X, no. 1, pp. 33–38, 2017.

[20] C. Y. Lin, "ROUGE : A Package for Automatic Evaluation of Summaries," in *Text Summarization Brances Out*, 2004, pp. 74–81.