Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.2025128860
Vol. 12, No. 1, Februari 2025, hlm. 69-76 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

PERBANDINGAN ALGORITMA STEMMING PORTER, SASTRAWI, IDRIS, DAN ARIFIN & SETIONO PADA DOKUMEN TEKS BAHASA INDONESIA

Jasman Pardede[*1], Dicky Darmawan[2]

[1,2]Institut Teknologi Nasional Bandung, Bandung
Email: *[1]jasman@itenas.ac.id, [2]darmawandicky59@mhs.itenas.ac.id
*Penulis Korespondensi

(Naskah masuk: 18 Maret 2024, diterima untuk diterbitkan: 10 Februari 2025)

Abstrak

Information Retrieval (*IR*) memiliki fungsi untuk memisahkan dokumen-dokumen yang relevan dari sekumpulan dokumen yang ada. Terdapat sebuah proses penting pada *IR*, yaitu *stemming*. *Stemming* adalah proses untuk mengurangi kata-kata berimbuhan menjadi bentuk kata dasar. Pada beberapa penelitian sering menggunakan *stemming* yang berbeda-beda, bahkan ada penelitian yang telah mencoba membandingkan dua algoritma *stemming*. Sedangkan pada penelitian ini membandingkan empat algoritma *stemming*. Tujuan penelitian ini adalah untuk mengungkapkan pengaruh *stemming* pada *IR* menggunakan dokumen bahasa Indonesia. Algoritma *stemming* yang digunakan pada penelitian ini adalah Porter, Sastrawi, Idris, dan Arifin Setiono. Kinerja masing-masing algoritma *stemming* diukur berdasarkan nilai presisi dan kebutuhan waktu. Terdapat 120 dokumen berbahasa Indonesia yang digunakan untuk mengukur kinerja masing-masing algoritma. Berdasarkan hasil eksperimen diperoleh bahwa kinerja terbaik untuk presisi adalah *stemming* Sastrawi. Sedangkan kinerja terbaik dari segi kebutuhan waktu adalah *stemming* Arifin Setiono. Kinerja presisi masing-masing *stemming* Sastrawi, Arifin Setiono, Porter, dan Idris adalah 70.3%, 55.8%, 49.9%, dan 32.2%. Sedangkan kinerja kebutuhan waktu rata-rata masing-masing *stemming* Arifin Setiono, Sastrawi, Idris, dan Porter adalah sebesar 0,123 detik, 160.1 detik, 168.8 detik, dan 188.4 detik.

Kata kunci: *Information Retrieval, Porter, Sastrawi, Idris, Arifin & Setiono*

COMPARISON OF STEMMING ALGORITHMS PORTER, SASTRAWI, IDRIS, AND ARIFIN SETIONO ON INDONESIAN TEXT DOCUMENTS

Abstract

Information Retrieval (IR) has the function of separating relevant documents from a set of existing documents. An important process in IR is stemming. Stemming is the process of reducing affixed words into basic word forms. In several studies, different stemming is often used, and even studies have tried to compare two stemming algorithms. In this study, four stemming algorithms were compared. The purpose of this study was to reveal the effect of stemming on IR using Indonesian language documents. The stemming algorithms used in this study were Porter, Sastrawi, Idris, and Arifin Setiono. The performance of each stemming algorithm was measured based on the accuracy value and time requirements. There were 120 Indonesian language documents used to measure the performance of each algorithm. Based on the experimental results, it was obtained that the best performance for precision was the Sastrawi stemming. While the best performance in terms of time requirements was the Arifin Setiono stemming. The precision performance of each stemming of Sastrawi, Arifin Setiono, Porter, and Idris is 70.3%, 55.8%, 49.9%, and 32.2%. Meanwhile, the average time requirement performance of each stemming of Arifin Setiono, Sastrawi, Idris, and Porter is 0.123 sec, 160.1 sec, 168.8 sec, and 188.4 sec.

Keywords: *Information Retrieval, Porter, Sastrawi, Idris, Arifin & Setiono*

1. PENDAHULUAN

Dalam diskusi yang membahas aspek-aspek pencarian informasi, terutama dalam konteks dokumen teks berbahasa Indonesia, *Information Retrieval* (*IR*) memiliki peran penting sebagai mekanisme untuk menyaring dan mengidentifikasi dokumen-dokumen yang dianggap relevan dari kumpulan besar dokumen yang tersedia. Proses ini melibatkan penggunaan algoritma dan teknik yang memungkinkan sistem *IR* untuk membedakan dokumen yang sesuai dengan kebutuhan pengguna (K. A. Hambarde & H. Proença, 2023).

Meningkatnya jumlah dokumen teks yang tersedia di internet, kebutuhan pengguna akan alat pencarian informasi yang efektif dan efisien juga bertambah. Keefektifan yang dimaksud adalah kinerja presisi temu balik dokumen yang bersesuaian dengan kueri yang diberikan dan kebutuhan waktu pencarian (Siswandi & Surojudin, 2020). Di dalam *IR*, terdapat bagian yang sangat penting yang disebut proses *stemming*. *Stemming* adalah proses untuk mengurangi kata-kata berimbuhan menjadi bentuk dasarnya (Rezalina, 2020).

Dalam konsep *stemming*, kata yang dipergunakan adalah bentuk umum yang akan dimasukkan ke dalam indeks (N. D. Arianti, dkk. 2019). Tujuannya adalah untuk menghasilkan dokumen yang lebih relevan. Proses *stemming* untuk Bahasa Indonesia melibatkan beberapa algoritma, seperti Arifin & Setiono tahun 2002, Jelita Asian tahun 2005, Ahmad Yusoff tahun 1996, Vega tahun 2001, Nazief & Adriani tahun 1996, Idris tahun 2001, Sastrawi tahun 2008, dan Porter tahun 1980. Setiap algoritma ini memiliki peran dalam mengubah kata-kata ke dalam bentuk dasarnya.

Dari sejumlah algoritma yang sudah diketahui, penelitian ini mempersempit pemilihan algoritma dengan memilih Porter, Sastrawi, Idris, dan Arifin & Setiono. Pemilihan keempat algoritma tersebut disebabkan oleh seringnya digunakan pada penelitian *IR* bahasa Indonesia. Selain itu terdapat algoritma yang memiliki tujuan tidak hanya untuk *stemming* bahasa Indonesia seperti algoritma Porter. Algoritma Porter merupakan algoritma untuk bahasa Inggris yang ditemukan pertama kali oleh Martin Porter pada tahun 1990 (Arif Siswandi, Permana & Emarilis, 2021). Algoritma Idris yang bertujuan untuk *stemming* bahasa Melayu dan dilakukan modifikasi terhadap teks bahasa Indonesia menjadi IN-Idris (Bahasa Indonesia) (Suci, Hayatin, & Munarko, 2022) serta Sastrawi dan Arifin & Setiono yang digunakan untuk bahasa Indonesia. Penelitian ini dilakukan untuk membandingkan keempat algoritma *stemming* dengan mengungkapkan *stemming* terbaik untuk bahasa Indonesia yang diukur berdasarkan efektivitas dan efisiensi. Efektivitas dan efisiensi dinilai menggunakan parameter kecepatan dan ketepatan dari masing-masing aturan algoritma yang berlaku (Sinaga & Nainggolan, 2023).

1.1 Rumusan Masalah

Berdasarkan pendahuluan penelitian ini merumuskan suatu masalah yaitu bagaimana hasil perbandingan keempat algoritma *stemming* dalam hal efektivitas dan efisiensi terhadap dokumen teks bahasa Indonesia.

1.2 Tujuan

Tujuan penelitian adalah untuk mengetahui efektivitas dan efisiensi algoritma *stemming* berdasarkan kinerja presisi dan kebutuhan waktu *stemming* dari algoritma Porter, Sastrawi, Idris dan Arifin & Setiono.

1.3 Lingkup Penelitian

Penelitian ini memiliki ruang lingkup penelitian yaitu, seluruh algoritma yang digunakan diuji terhadap dokumen teks bahasa Indonesia. Seluruh algoritma *stemming* dievaluasi dengan dokumen uji yang sama untuk mengungkap kinerja masing-masing algoritma *stemming* berdasarkan efektivitas dan efisiensi.

2. METODE PENELITIAN

2.1 *Stemming*

Algoritma *stemming* ialah suatu teknik yang digunakan untuk meningkatkan kinerja *IR* dengan mengubah kata-kata dalam dokumen teks ke bentuk kata dasar (E. Colmenares & H. Wu, 2023). Dengan menerapkan proses *stemming*, efisiensi algoritma *IR* dapat ditingkatkan karena menghapus imbuhan kata-kata yang memiliki variasi morfologis tetapi memiliki makna semantik yang sama (Wisuda Sardjono et al., 2020). Penerapan algoritma *stemming* pada bahasa Indonesia menjadi rumit karena adanya berbagai variasi dan kombinasi imbuhan yang harus dihilangkan untuk memperoleh kata dasar. Dalam penelitian ini, digunakan empat algoritma *stemming*, yaitu Porter, Sastrawi, Idris, dan Arifin Setiono, untuk bahasa Indonesia. Berikut adalah penjelasan singkat tentang setiap algoritma tersebut.

2.2 Algoritma Porter

Algoritma Porter *Stemmer* merupakan suatu metode untuk mengenali kata dasar dengan menghilangkan afiks atau imbuhan (A. D. Hartanto. et al., 2023). Proses ini dapat menyebabkan kebingungan atau ambiguitas makna pada kata-kata karena ketidak konsistenan dalam aturan morfologi bahasa Indonesia (Permana, 2017). Algoritma Porter diusulkam oleh Martin Porter. *Stemming* Porter melakukan proses membuang kata awalan secara berulang, sesuai dengan kaidah bahasa Inggris. *Stemming* Porter mengalami beberapa modifikasi aturan untuk bahasa Indonesia. Pada tahun 1992 W.B Frakes berhasil memodifikasi dan mengembangkan aturan Porter bahasa Inggris menjadi bahasa Indonesia (Rahmatulloh et al., 2020). Tahapan algoritma Porter meliputi:

1. Menghapus partikel (“lah”, “kah”, “tah”, “pun”).
2. Menghapus kata ganti (*Possessive Pronoun*), seperti –ku, -mu, -nya
3. Menghapus awalan pertama. Jika tidak ditemukan, maka lanjut ke langkah 4*a*, dan jika ada maka lanjut ke langkah 4*b*
4. *a*. Menghapus awalan kedua, dan dilanjutkan pada langkah ke 5*a*

b. Menghapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai kata dasar *(root word)*. Jika ditemukan maka lanjut ke langkah 5*b*.

5. *a*. Menghapus akhiran dan kata akhir diasumsikan sebagai kata dasar *(root word)*.
   b. Menghapus awalan kedua dan kata akhir diasumsikan sebagai kata dasar. Dalam sebuah kata, memungkinkan adanya dua awalan yang saling berurutan.

Beberapa aturan urutan awalan yang diperbolehkan dalam algoritma Porter dapat dilihat pada Tabel 1 (Wahyudi, Susyanto & Nugroho, 2017).

Tabel 1. Awalan Kata yang Diperbolehkan

Awalan1	Awalan2
Meng	Per
Di	Ber
Ter	
Ke	

2.3 Algoritma Sastrawi

Pustaka Sastrawi merupakan sebuah sumber daya yang digunakan untuk memproses kata-kata agar dapat diubah menjadi bentuk dasarnya (B. Siswanto & Y. Dani, 2021). Sastrawi *stemmer* mengikuti prinsip yang dikembangkan oleh Nazief dan Adriani, dan telah diperbaiki dengan algoritma *CS* (*Confix Stripping*), Algoritma *ECS* (*Enhanced Confix Stripping*), serta dimodifikasi lebih lanjut dengan Modifikasi *ECS* (Rosid et al., 2020). Berikut adalah langkah-langkah dalam pembuatan algoritma Sastrawi.

1. Melakukan pemeriksaan apakah kata yang akan di *stemming* ada dalam kamus kata dasar atau tidak. Jika ada, maka proses *stemming* berhenti.
2. Jika tidak ada dalam kamus, artinya kata tersebut merupakan kata berimbuhan. Kemudian menghilangkan kata akhiran "-lah", "-kah", "-ku", "-mu", "-nya", "-tah" atau "-pun".
3. Menghilangkan kata imbuhan akhiran "-i","-kan","-an", kemudian hapus kata imbuhan awalan "be-", "di-", "ke-", "me-", "pe-", "se-", dan "te-".
4. Jika kata dasar yang dihasilkan dari langkah sebelumnya tidak terdapat di kamus, maka kata tersebut dicek apakah termasuk pada tabel keambiguan atau tidak.
5. Jika seluruh proses langkah 1-4 gagal dilakukan, maka algoritma mengembalikan kata aslinya.

2.4 Algoritma Idris

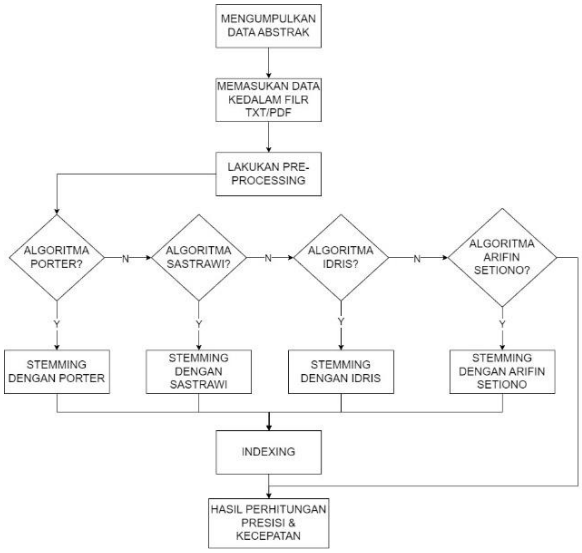
Algoritma Idris juga menyertakan serangkaian langkah-langkah untuk melakukan proses penentuan kata dasar pada teks dokumen dalam bahasa Indonesia. Berikut adalah langkah-langkah algoritma Idris untuk mendapatkan kata dasar pada teks dokumen bahasa Indonesia. (Suci, Hayatin & Munarko, 2022).

1. Kata yang belum di-*stemming* dicari pada kamus umum atau Kamus Besar Bahasa Indonesia (KBBI). Jika kata tersebut ditemukan, berarti kata tersebut adalah kata dasar. Kata tersebut dikembalikan dan algoritma dihentikan.
2. Kata yang belum di-*stemming* dicari pada kamus lokal. Jika kata tersebut langsung ditemukan, berarti kata tersebut adalah kata dasar. Kata tersebut dikembalikan dan algoritma dihentikan. Kamus lokal digunakan karena algoritma Idris dibuat untuk *stemming* bahasa Melayu, yang mana kamus lokal berisi kata-kata dasar bahasa Melayu. Sedangkan *stemming* bahasa Indonesia tidak memerlukan kamus lokal.
3. Menghilangkan turunan awalan. Langkah ini terus dilakukan sampai tidak ada lagi turunan awalan. Jika tidak ada lagi, maka lanjutkan ke langkah berikutnya. Untuk beberapa kombinasi imbuhan, dilakukan penghilangan awalan terlebih dahulu, yaitu pada kombinasi imbuhan "ber-lah", "ber-an", "men-i", "di-i", "pe-i", "ter-i".
4. Hilangkan infleksi akhiran terlebih dahulu. Jika hal ini berhasil dan akhiran adalah partikel ("-lah", "-pun", dan "-kah"), langkah ini dilakukan lagi untuk menghilangkan kata ganti akhiran posesif ("ku", "mu", atau "nya").
5. Hilangkan turunan akhiran. Langkah ini terus dilakukan sampai tidak ada lagi penurunan akhiran. Jika tidak ada lagi, maka lanjutkan ke langkah berikutnya.
6. Setelah setiap penghilangan imbuhan dilakukan, maka lakukan pengecekan menggunakan kamus. Jika kata ditemukan, maka algoritma berhenti dan tidak perlu dilakukan pengecekan terhadap imbuhan lainnya. Jika sampai selesai penghilangan imbuhan masih belum menemukan kata dasar, maka dilakukan *recording*.
7. Jika semua langkah sudah dilakukan termasuk *recoding* dan tidak juga ditemukan dalam kamus, maka algoritma ini akan menganggap kata semula sebagai kata dasar.

2.5 Algoritma Arifin & Setiono

Menurut (Jatikusumo & Derajad Wijaya, 2021) algoritma Arifin & Setiono melakukan pembacaan tiap kata yang ada terlebih dahulu, sehingga mendapatkan tahapan – tahapan yang dilakukan algoritma ini sebagai berikut:

1. Pemeriksaan semua kemungkinan bentuk kata. Setiap kata diasumsikan memiliki 2 awalan (prefiks) dan 3 akhiran (sufiks). Jika dalam kata yang diperiksa tidak memiliki imbuhan sebanyak imbuhan seperti formula di atas, maka imbuhan yang kosong atau tidak ada tersebut diberi tanda ***x*** untuk prefiks dan diberi tanda ***xx*** untuk sufiks.



Gambar 1. Alur Teknik Penelitian

2. Pemotongan dalam algoritma ini dilakukan secara berurutan sebagai berikut:
   AW : AW (Awalan)
   AK : AK (Akhiran)
   KD : KD (Kata Dasar)
   a. AWI, hasilnya disimpan ke pe1(*prefiks 1*)
   b. AWII, hasilnya disimpan ke pe2(*prefiks 2*)
   c. AKI, hasilnya disimpan ke su1(*sufiks 1*)
   d. AKII, hasilnya disimpan ke su2(*sufiks 2)*
   e. AKIII, hasilnya disimpan ke su3(*sufiks 3*)
   Dalam setiap tahap pemotongan di atas selalu diikuti dengan pemeriksaan di dalam kamus. Hal ini untuk mengetahui apakah hasil pemotongan tersebut sudah ada dalam bentuk dasar. Apabila pemeriksaan ini berhasil maka proses dinyatakan selesai dan tidak perlu melanjutkan proses pemotongan imbuhan selanjutnya.
3. Akan tetapi, apabila sampai pada pemotongan AK III, belum ditemukan dalam kamus, maka dilakukan proses kombinasi. Kata dasar yang dihasilkan dikombinasikan dengan imbuhan-imbuhannya dalam 12 konfigurasi berikut :
   a. KD
   b. KD + AK III
   c. KD + AK III + AK II
   d. KD + AK III + AK II + AK I
   e. AW I + AW II + KD
   f. AW I + AW II + KD + AK III
   g. AW I + AW II + KD + AK III + AK II
   h. AW I + AW II + KD + AK III + AKII + AKI
   i. AW II + KD
   j. AW II + KD + AK III
   k. AW II + KD + AK III + AK II
   l. AW II + KD + AK III + AK II + AK I

Hasil kombinasi *a*, *b*, *c*, *d*, *h*, dan *l* telah diuji pada tahap sebelumnya, karena merupakan hasil dari pemotongan bertahap. Oleh karena itu, fokus hanya pada kombinasi-kombinasi yang belum diuji (*e, f, g, i, j,* dan *k*). Jika terdapat dalam proses kombinasi yang dilakukan, maka pemeriksaan pada kombinasi lainnya tidak lagi diperlukan. Pemeriksaan pada 12 kombinasi ini penting karena adanya fenomena *overstemming* dalam algoritma.

2.6 Teknik Penelitian

Metode penelitian yang digunakan mencakup pengumpulan dokumen teks yang mengandung kata-kata berimbuhan. Sebanyak 120 dokumen dalam bahasa Indonesia diuji, dengan kriteria minimal 50 kata berimbuhan dan maksimal 3000 kata berimbuhan. Hal ini dibatasi karena keterbatasan waktu eksekusi maksimal bahasa pemrograman yang terbatas yang digunakan peneliti. Meskipun demikian, semakin banyak kata yang mengalami proses *stemming*, diharapkan hasilnya semakin optimal. Setelah dokumen-dokumen terkumpul, dilakukan proses unggah pada masing-masing algoritma *stemming* tanpa melakukan kombinasi, Adapun alur penelitian yang dilakukan adalah seperti yang dinyatakan pada Gambar 1.

2.7 Algoritma Presisi

Dari langkah – langkah yang dijelaskan pada bagian teknik penelitian, nilai presisi algoritma dihitung menggunakan persamaan (1).

$$Presisi = \frac{RW}{W} x\ 100 \qquad (1)$$

Dimana $\boldsymbol{W}$ adalah jumlah kata yang dilakukan proses *stemming*, dan $\boldsymbol{RW}$ adalah jumlah kata yang berhasil dilakukan *stemming* atau dinyatakan proses *stemming*-nya benar. Serta perhitungan nilai presisi ini akan dinyatakan dalam bentuk persen (%).

Sebagai contoh penggunaan algoritma Porter pada suatu studi kasus yang memiliki 10 data. Banyaknya kata yang tepat di *stemming* menggunakan algoritma Porter yang dihitung secara manual adalah 6 kata. Sehingga, nilai presisinya adalah 60%, seperti pada contoh Gambar 2.

No	Kata	Jumlah	Status
1	Cari	2	Berhasil
2	Informasi	1	Berhasil
3	upa	1	Gagal
4	kenal	1	Berhasil
5	istilah	1	Berhasil
6	information	1	Gagal
7	retrieval	1	Gagal
8	rupa	1	Berhasil
9	pisah	1	Berhasil
10	anggap	1	Berhasil
	jumlah berhasil	6	
	jumlah kata	10	
	presisi	(6/10) *100 =	60 %

Gambar 2. Algoritma presisi

2.8 Rumus Perhitungan Waktu

Dalam penelitian ini, pengukuran parameter waktu digunakan untuk mengevaluasi kecepatan algoritma yang diterapkan. Perhitungan waktu ini diimplementasikan melalui pemrograman PHP dengan menggunakan fungsi *microtime*. Proses ini dilakukan dengan memasang *microtime* pada awal eksekusi algoritma dan juga pada saat algoritma selesai dijalankan. Besarnya waktu yang diperlukan untuk mendapatkan kata dasar dari setiap dokumen dinyatakan dengan menggunakan persamaan (2).

$$Waktu = EndMicrotime - StartMicrotime \quad (2)$$

Pada persamaan (2) *StartMicrotime* digunakan saat awal algoritma dijalankan agar mengatur awal mula algoritma bekerja, sedangkan *EndMicrotime* digunakan pada akhir algoritma dijalankan agar menghitung waktu terakhir yang ditempuh oleh algoritma tersebut. Setelah itu akan dilakukan pengurangan antara *EndMicrotime* dengan *StartMicrotime* agar hasilnya menjadi detik (PHP 2023).

2.9 Blok Diagram

Pada penelitian ini mengikuti beberapa proses. Setiap proses temu balik dokumen teks bahasa Indonesia yang diimplementasikan mengikuti proses yang sama, baik pada *stemming* Porter, Sastrawi, Idris, dan Arifin Setiono. Sehingga perbedaan kinerja diperhatikan berdasarkan proses *stemming* yang digunakan. Adapun blok diagram sistem penelitian adalah seperti yang dinyatakan pada Gambar 3.

Pada blok diagram sistem yang dinyatakan pada Gambar 3 mengikuti tahapan-tahapan sebagai berikut:

1. Proses pertama adalah melakukan *input* data atau *upload* dokumen berekstensi *.txt* kedalam sistem
2. Lakukan inisialisasi awal waktu eksekusi (*StartMicrotime*) proses *stemming* menggunakan fungsi *microtime php*
3. Lakukan *pre-processing* terdapat dokumen yang telah di *upload*
4. Menentukan kata dasar dengan menggunakan masing-masing algoritma *stemming* dari hasil *pre-processing*
5. Sistem mengembalikan hasil proses *stemming* setiap dokumen.
6. Tentukan waktu akhir yang diperoleh setelah proses *stemming*. Hitung waktu eksekusi *stemming* dengan menggunakan *EndMicrotime*.
7. Dari hasil *EndMicrotime* diperoleh waktu eksekusi *stemming* menggunakan persamaan (2)
8. Setiap kata dasar yang dihasilkan dari proses *stemming* disimpan kedalam *indexing*
9. Tentukan ketepatan hasil proses *stemming* untuk setiap algoritma dengan menggunakan persamaan (1)
10. Langkah terakhir adalah *user* diminta memasukkan data ke dalam *database* dengan menekan tombol *submit* (jika ingin menyimpan *history* hasil dari proses *stemming*).

3. HASIL DAN PEMBAHASAN

Dalam penelitian ini, pengujian dilakukan dengan cara pengguna memasukkan dokumen teks berbahasa Indonesia ke dalam sistem sesuai dengan algoritma yang dipilih. Adapun proses penggunaan data dan pengujian sistem seperti yang dijelaskan berikut ini.

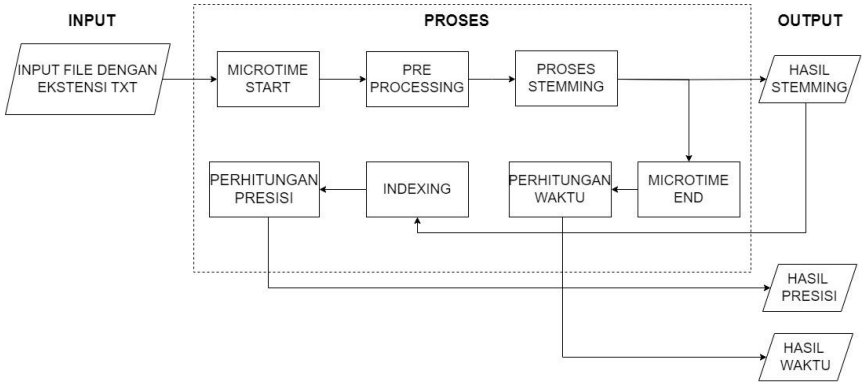
3.1 Penggunaan Data

Dalam penelitian ini, *dataset* terdiri dari 120 dokumen teks berbahasa Indonesia dalam format *.pdf* atau *.txt*. Setiap dokumen teks memiliki setidaknya 50 kata berimbuhan dan maksimal 3000 kata berimbuhan. Hal ini dilakukan untuk membatasi eksekusi program terhadap dokumen-dokumen tersebut. Meskipun demikian, semakin banyak kata berimbuhan yang digunakan, diharapkan kinerja dari setiap algoritma juga semakin optimal.

3.2 Pengujian Sistem

Dalam pengujian sistem ini, dilakukan validasi terhadap algoritma yang sudah ada dan algoritma yang dikembangkan khusus untuk penelitian ini. Sebagai contoh, dokumen yang digunakan adalah dokumen *kata-berimbuhan.txt* untuk membandingkan antara penggunaan *library* dan tanpa penggunaan *library*, yang menghasilkan tingkat ketepatan *library* sebesar 75% dan tanpa *library* sebesar 80%. Dapat disimpulkan bahwa algoritma yang dikembangkan tanpa menggunakan *library* memiliki hasil yang lebih baik daripada menggunakan *library*.

Perbedaan hasil tersebut disebabkan oleh perbedaan jumlah kata dasar yang terdapat dalam kamus pada *library*, yang tidak komprehensif yang dimiliki pada saat penelitian dilakukan. Beberapa contoh kata-kata yang tidak berhasil di-*stemming* oleh algoritma yang digunakan dalam *library* adalah seperti yang dinyatakan pada Tabel 2.



Gambar 3. Blok Diagram

Tabel 2. Hasil Perbedaan *Stemming* dengan *Library* & tanpa *Library*

Kata Berimbuhan	Hasil Library	Hasil Tanpa Library
Mengerikan	Mengerikan	Ngeri
Memainkan	Memainkan	Main
Menasihatin	Menasihati	Nasihat
Memudahkan	Memudahkan	Mudah
Memanipulasi	Memanipulasi	Manipulasi
Menari	Menari	Tari
Memulai	Memulai	Mulai
Memakan	Memakan	Makan
Memaafkan	Memanfaatkan	Manfaat

3.3 Hasil *Stemming*

Dalam penelitian ini, diperoleh hasil dari proses *stemming* yang menunjukkan nilai ketepatan dan kecepatan dari masing-masing algoritma dalam mengurangi kata berimbuhan menjadi kata dasar pada 120 data yang diuji. Semua kata dasar yang diuji akan dijelaskan dalam Tabel 3 untuk ketepatan dan Tabel 4 untuk kecepatan, sehingga penjelasan dan hasil dapat lebih jelas terlihat untuk setiap perbedaan algoritma.

Tabel 3. Hasil Perhitungan Ketepatan *Stemming*

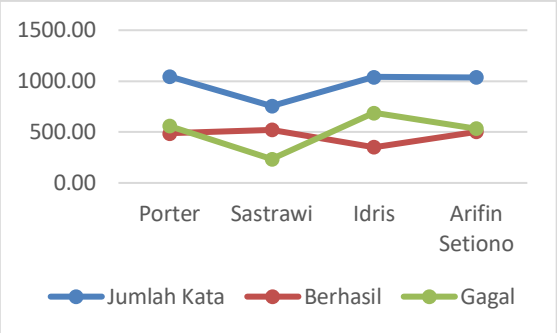
Dokumen	Porter (%)	Sastrawi (%)	Idris (%)	Arifin & Setiono (%)
stemming.txt	64.00	97.00	43.65	65.87
microsleep.pdf	42.91	75.09	29.68	52.16
Perbandingan algoritma stemming sastrawi.pdf	51.17	76.30	34.13	55.12
Document61.txt	51.39	79.83	31.81	49.14
Document63.txt	40.11	79.65	27.57	51.41
document67.txt	53.15	79.22	36.38	47.52
document72.txt	50.23	78.35	34.21	51.99
document80.txt	47.77	78.48	34.82	55.64
document66.txt	38.08	78.49	26.96	54.48
Kata Berimbuhan.txt	60.28	82.00	32.90	75.31

Pada Tabel 3 memperlihatkan hasil proses *stemming* masing-masing algoritma untuk 10 dokumen uji dari 120 dokumen uji. Pada dokumen *stemming.txt* kinerja presisi Porter, Sastrawi, Idris, dan Arifin & Setiono masing-masing adalah 64.00%, 97.00%, 43.65%, dan 65.87%.

Tabel 4. Hasil Perhitungan Kecepatan *Stemming*

Dokumen	Porter (detik)	Sastrawi (detik)	Idris (detik)	Arifin & Setiono (Detik)
stemming.txt	51.48	40.98	53.03	0.016
microsleep.pdf	211.3	181.3	151.8	0.167
Perbandingan algoritma stemming sastrawi.pdf	237.2	163.1	213.3	0.173
Document61.txt	130.7	120.8	117.3	0.116
Document63.txt	340.5	277.1	311.6	0.137
document67.txt	152.0	130.6	133.4	0.134
document72.txt	129.9	110.0	108.1	0.123
document80.txt	193.4	149.4	154.5	0.125
document66.txt	155.6	120.2	132.3	0.111
Kata Berimbuhan.txt	168.4	210.1	180.2	0.123

Pada Tabel 4 memperlihatkan kebutuhan waktu proses *stemming* untuk 10 dokumen uji dari 120 dokumen uji yang digunakan. Setiap dokumen uji dihitung waktu eksekusi proses *stemming* setiap algoritma. Pada dokumen uji *stemming.txt* waktu eksekusi yang diperlukan untuk melakukan *stemming* pada algoritma Porter, Sastrawi, Indris, dan Arifin Setiono masing-masing adalah 51.48 detik, 40.98 detik, 53.03 detik, dan 0.016 detik. Hal yang sama untuk dokumen uji lainnya. Sedangkan perbandingan rata-rata jumlah kata yang berhasil dilakukan proses *stemming* dengan jumlah kata yang gagal pada setiap algoritma dinyatakan pada Gambar 4.



Gambar 4. Kinerja rata-rata hasil *stemming*

Berdasarkan hasil eksperimen untuk 120 dokumen uji diperoleh bahwa kinerja terbaik untuk presisi adalah *stemming* Sastrawi. Sedangkan kinerja terbaik dari segi kebutuhan waktu adalah *stemming* Arifin Setiono. Kinerja presisi masing-masing *stemming* Sastrawi, Arifin Setiono, Porter, dan Idris adalah 70.3%, 55.8%, 49.9%, dan 32.2%. Sedangkan kinerja kebutuhan waktu rata-rata masing-masing *stemming* Arifin Setiono, Sastrawi, Idris, dan Porter adalah sebesar 0,123 detik, 160.1 detik, 168.8 detik, dan 188.4 detik.

3.4 Analisis

Setelah berhasil menyelesaikan semua tahapan pengujian, baik itu pengujian sistem maupun pengujian kinerja metode atau algoritma, penelitian ini melakukan perbandingan kinerja seluruh algoritma dengan mengukur nilai ketepatan dan kecepatan dari masing-masing algoritma yang digunakan, termasuk Porter, Sastrawi, Idris, dan Arifin Setiono. Nilai yang dihasilkan dipengaruhi oleh jumlah kata berimbuhan dalam file dengan ekstensi *.txt* atau *.pdf*. Keberhasilan *stemming* juga dipengaruhi oleh langkah-langkah yang diimplementasikan dalam setiap algoritma. Dengan demikian, dapat dikatakan bahwa semakin banyak kata yang berhasil melalui proses *stemming*, semakin tinggi pula nilai ketepatan dari algoritma yang digunakan. Namun, semakin kompleks aturan dalam algoritma, proses eksekusi program akan memakan waktu lebih lama. Proses *stemming* ini telah melewati tahapan validasi dengan menggunakan *library stemming* yang sudah ada. Hasil dari algoritma yang dibuat tanpa menggunakan *library* juga lebih optimal dibandingkan dengan hasil menggunakan *library*, karena perbedaan dalam cakupan kamus bahasa Indonesia yang dimiliki oleh *library* yang jumlahnya lebih rendah dibandingkan dengan algoritma tanpa *library*.

Dilihat dari rekapitulasi kinerja sistem dalam Tabel 3 dan Tabel 4, serta dari grafik hasil pengujian, terlihat data mengenai file-file yang digunakan untuk proses *stemming*. Dari data tersebut, dapat disimpulkan bahwa algoritma Sastrawi memiliki nilai ketepatan yang lebih tinggi dibandingkan dengan algoritma Arifin Setiono. Namun, algoritma Arifin Setiono memiliki nilai yang lebih tinggi dibandingkan dengan algoritma Porter. Algoritma Porter memiliki nilai yang lebih tinggi dibandingkan dengan algoritma Idris.

Namun, dari segi perhitungan waktu yang dinyatakan pada Tabel 3, menyatakan bahwa data waktu algoritma Arifin Setiono lebih efisien dibandingkan dengan algoritma lainnya. Berdasarkan hasil eksperimen yang telah disusun, terlihat bahwa jumlah kata terbanyak terdapat pada *dokumen63.txt* dengan jumlah 2633 kata, sementara kata terendah hanya terdapat 126 kata pada dokumen *stemming.txt*. Menurut (Maulipaksi, 2016), jumlah kata dasar yang ada dalam KBBI mencapai 41.472 kata. Dalam penelitian ini, berhasil melakukan proses *stemming* pada kata dasar sebanyak 34.402 kata menggunakan keempat algoritma yang dikembangkan. Artinya terdapat 17.05% kata dasar yang masih belum berhasil dilakukan proses *stemming*.

4. KESIMPULAN

Setelah melakukan uji coba pada sistem yang dikembangkan dalam penelitian ini, dapat disimpulkan bahwa penggunaan kata dalam dokumen minimal sebanyak 50 dan maksimal sebanyak 3000 kata memberikan hasil pengujian sistem yang optimal. Setiap langkah dalam algoritma memiliki dampak signifikan terhadap durasi eksekusi dan mempengaruhi keberhasilan *stemming*. Secara keseluruhan, algoritma Sastrawi menunjukkan kinerja lebih baik dibandingkan dengan algoritma lainnya, sementara algoritma Arifin Setiono mencapai waktu eksekusi tercepat dibandingkan dengan algoritma *stemming* lainnya. Penelitian ini berhasil mereduksi kata dasar sebanyak 34.402 dari total 41.472 kata dasar yang terdapat dalam kamus besar bahasa Indonesia.

DAFTAR PUSTAKA

A. D. HARTANTO, Y. PRISTYANTO, A. N. ROHMAN, E. PUJASTUTI, A. NURMASANI & I. A. ASTUTI. 2023. Measuring of Scientific Document Abstraction Similarity Using Rabin-Karp and Poter Stemmer. International Conference on Informatics, Multimedia, Cyber and Informations System (ICIMCIS), Jakarta Selatan, Indonesia, 2023, pp. 49-54.<https://doi.org/10.1109/ICIMCIS60089.2023.10348988>

ARIF SISWANDI, A., YUDI PERMANA & ARVITA EMARILIS. 2021.Stemming Analysis Indonesian Language News Text with Porter Algorithm. In: . Journal of Physics: Conference Series. Vol. 1845. IOP Publishing Ltd. <https://doi.org/10.1088/1742-6596/1845/1/012019>

B. SISWANTO & Y. DANI. 2021. Sentiment Analysis about Oximeter as Covid-19

Detection Tools on Twitter Using Sastrawi Library. 8th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE), Semarang, Indonesia, 2021, pp. 161-164. <https://doi.org/10.1109/ICITACEE53184.2021.9617216>

E. COLMENARES & H. WU. 2021. Accelerating Workload Processing with MPI for Porter's Stemming Algorithm. 2021 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2021, pp. 1783-1787. <https://doi.org/10.1109/CSCI54926.2021.00337>

JATIKUSUMO, DWIKI & HERRY DERAJAD WIJAYA. 2021. IJCIT (Indonesian Journal on Computer and Information Technology) Pendeteksi Lokasi Kejadian Covid-19 Menggunakan Social Media Dengan Kombinasi Algoritma Stemming Bahasa Indonesia. *IJCIT (Indonesian Journal on Computer and Information Technology)*. Vol. 6. <https://doi.org/10.31294/ijcit.v6i1>

K. A. HAMBARDE & H. PROENÇA. 2023 Information Retrieval: Recent Advances and Beyond, in IEEE Access, vol. 11, pp. 76581-76604, <https://doi.org/10.1109/ACCESS.2023.3295776>.

MAULIPAKSI, DESLIANA. 2016. Badan Bahasa Kemendikbud Luncurkan KBBI Edisi IV Daring. Kementrian Pendidikan Dan Kebudayaan. July 27, 2016. <https://www.kemdikbud.go.id/main/blog/2016/07/badan-bahasa-kemendikbud-luncurkan-kbbi-edisi-iv-daring>

N. D. ARIANTI, M. IRFAN, U. SYARIPUDIN, D. MARIANA, N. ROSMAWARNI & D. S. MAYLAWATI. 2019. Porter Stemmer and Cosine Similarity for Automated Essay Assessment. 2019 5th International Conference on Computing Engineering and Design (ICCED), Singapore, 2019, pp. 1-6. < https://doi.org/10.1109/ICCED46541.2019.9161090>.

PERMANA, A YUDI. 2017. Implementasi Stemming Porter KBBI Untuk Klasifikasi Topik Soal Ujian Nasional Bahasa Indonesia Menggunakan Algoritma Naive Bayes. *Jurnal Teknologi Pelita Bangsa*. <https://doi.org/10.37366/sigma.v8i3.126>

RAHMATULLOH, ALAM, NENG IKA KURNIATI, IRFAN DARMAWAN, ADI ZAENAL ASYIKIN & DEDEN WITARSYAH. 2020. Comparison of the Effects Stemmer Porter and Nazief-Adriani on the Performance of Winnowing Algorithms for Measuring Plagiarism. *Journal of Digital Information Management* 18: 49. <https://doi.org/10.6025/jdim/2020/18/2/49-56>.

REZALINA, OPPIE. 2020. Perbandingan Algoritma Stemming Nazief & Adriani, Porter Dan Arifin Setiono Untuk Dokumen Teks Bahasa Indonesia. Universitas muhammadiyah jember, <http://repository.unmuhjember.ac.id/550/1/JURNAL.pdf >.

ROSID, MOCHAMAD ALFAN, ARIF SENJA FITRIANI, IKA RATNA INDRA ASTUTIK, NASRUDIN IQROK MULLOH & HARIS AHMAD GOZALI. 2020. Improving Text Preprocessing for Student Complaint Document Classification Using Sastrawi. In: . *IOP Conference Series: Materials Science and Engineering*. Vol. 874. Institute of Physics Publishing. <https://doi.org/10.1088/1757-899X/874/1/012017>.

SINAGA, ARDILES & SAHAT PANDAPOTAN NAINGGOLAN. 2023. Analisis Perbandingan Akurasi Dan Waktu Proses Algoritma Stemming Arifin-Setiono Dan Nazief-Adriani Pada Dokumen Teks Bahasa Indonesia. *Sebatik* 27: 63–69. <https://doi.org/10.46984/sebatik.v27i1.2072>.

SISWANDI, ARIF & NURHADI SUROJUDIN. 2020. Analisis Dan Perbandingan Stemming Algoritma Porter Dengan Algoritma Ahmad Yusoff Sembok Dalam Dokumen Teks Bahasa Indonesia. *Seminar Nasional Teknologi Informasi Dan Komunikasi STI&K (SeNTIK)* 4. <https://ejournal.jak-stik.ac.id/files/journals/2/articles/sentik2020/324/submission/proof/324-13-1121-1-10-20201101.pdf>

SUCI, FEBIARTY WULAN, NUR HAYATIN & YUDA MUNARKO. 2022. In-Idris: Modification Of Idris Stemming Algorithm For Indonesian Text. *IIUM Engineering Journal* 23: 82–94. <https://doi.org/10.31436/IIUMEJ.V23I1.1783>.

WAHYUDI, DWI, TEGUH SUSYANTO & DIDIK NUGROHO. 2017. Implementasi Dan Analisis Algoritma Stemming Nazief & Adriani Dan Porter Pada Dokumen Berbahasa Indonesia. Jurnal Ilmiah SINUS, 15 (2). <https://doi.org/10.30646/sinus.v15i2.305 >

WISUDA SARDJONO, MOCHAMMAD, MARGI CAHYANTI, MAULANA MUJAHIDIN & RINI ARIANTY. 2020. Pendeteksi Kesamaan Kata Untuk Judul Penulisan Berbahasa Indonesia Menggunakan Algoritma Stemming Nazief-Adriani. <https://jurnal.wicida.ac.id/index.php/sebatik/article/view/320>