ISSN (Print) : 1693-1173
ISSN (Online) : 2548-4028

IMPLEMENTASI DAN ANALISIS ALGORITMA STEMMING NAZIEF & ADRIANI DAN PORTER PADA DOKUMEN BERBAHASA INDONESIA

Dwi Wahyudi[1)], Teguh Susyanto[2)], Didik Nugroho[3)]
[1,3)]Program Studi Teknik Informatika, STMIK Sinar Nusantara Surakarta
[2)]Program Studi Sistem Informasi, STMIK Sinar Nusantara Surakarta
[1)]wahyudi452@gmail.com, [2)]teguh@sinus.ac.id, [3)]masdidikhoho@sinus.ac.id

Abstract

Stemming is a process to return a derivative word into its root word by eliminating the affixes. This is necessary to support a better information retrieval system. Some research on stemming algorithm, including algorithm Nazief & Adriani and Porter. Each stemmer has advantages and disadvantages of each. The purpose of this study is to compare the two stemmers, so it is known which algorithm is better to support information retrieval system. This research mostly applied literature study with reference to research conducted by Asian Jelita and Fadillah Z Tala. Test documents obtained from online news sites (detik.com). The process of analysis is done by calculating the number of correct results and experiencing stemming errors (overstemming, understemming, unchanged, spelling exception), then comparing the result and time of the process, so it will be known which stemmer is better to support information retrieval system.

Keywords: *Information retrieval, Stemming, Nazief & Adriani algorithm, Porter algorithm*

I. PENDAHULUAN

Dalam sistem temu kembali informasi (*Information Retrieval*), algoritma *stemming* digunakan untuk mengurangi perbedaan bentuk dari suatu kata dengan mengembalikannya ke dalam bentuk kata dasar sehingga proses temu kembali menjadi efektif (Asian, Williams, & Tahaghoghi, 2005).

Stemming adalah proses pemotongan (pembuangan) imbuhan (*affix*), baik *prefix* maupun *suffix*, dari sebuah term untuk mendapatkan kata dasar (*root* atau *stem*) dari kata berimbuhan. Beberapa algoritma *stemming* untuk Bahasa Indonesia telah dikembangkan sebelumnya, diantaranya algoritma Nazief & Adriani (Asian, 2007) dan algoritma Porter (Tala, 2003). Algoritma Nazief dan Adriani dikembangkan berdasarkan aturan morfologi Bahasa Indonesia yang mengelompokkan imbuhan menjadi awalan (*prefix*), sisipan (*infix*), akhiran (*suffix*) dan gabungan awalan akhiran (*confixes*). Algoritma ini menggunakan kamus kata dasar dan mendukung *recoding*, yakni penyusunan kembali kata-kata yang mengalami proses *stemming* berlebih. Kamus kata dasar tersebut dibutuhkan untuk memeriksa apakah kata dasar yang melalui proses *stemming* benar dan ditemukan pada kamus saat proses *stemming* dilakukan. Algoritma Porter menghilangkan imbuhan-imbuhan yang ada berdasarkan aturan morfologi dalam Bahasa Indonesia, tanpa menggunakan kamus. Masing-masing *stemmer* memiliki kelebihan dan kekurangannya masing-masing. Efektifitas algoritma *stemming* dapat diukur berdasarkan beberapa parameter, seperti kecepatan proses, keakuratan, dan kesalahan.

Penelitian terkait *stemming* bahasa indonesia pernah diusulkan beberapa peneliti. *Stemming* bahasa indonesia dengan algoritma Nazief & Andriani diusulkan oleh (Marsya & Abidin, 2011), adapun hasil yang diperoleh bahwa *stemming* sufiks terlebih dahulu pada suatu kata memiliki hasil yang lebih baik dibanding *stemming* afiks, dan pemotongan sufiks "-an" diikuti dengan pemotongan sufiks "-kan" lebih baik dibandingkan pemotongan sufiks "-kan". Pada penelitian (Afuan, 2013) mengusulkan sebuah *stemming* bahasa

indonesia dengan menggunakan algoritma Porter tanpa menggunakan kamus. Kemudian penelitian (Utomo, 2013) memanfaatkan algoritma *stemmer* tala untuk melakukan *stemming* bahasa indonesia. Proses *stemming* memanfaatkan *rule base*. Pada proses *stemming* ini didapatkan performa relatif stabil namun memiliki tingkat kesalahan yang relatif tinggi.

Penelitian ini dimaksudkan untuk membandingkan algoritma Nazief & Adriani dengan algoritma Porter untuk proses *stemming* pada teks ber-Bahasa Indonesia, sehingga akhirnya akan diketahui algoritma manakah yang lebih cepat, lebih akurat atau yang lebih banyak melakukan kesalahan *stemming*.

II. TINJAUAN PUSTAKA

2.1. *Information Retrieval*

Information retrieval pada dasarnya merupakan proses untuk menentukan dokumen dalam koleksi yang harus ditemubalikkan untuk memenuhi keinginan pengguna akan informasi.

Suatu sistem temu balik informasi dikatakan ideal jika sistem tersebut dapat menemukan seluruh dokumen yang relevan dan sistem hanya menemukan dokumen yang relevan saja. Banyaknya varian morfologik pada suatu bahasa sangat berpengaruh terhadap hal tersebut, untuk itu setiap kata harus diubah kedalam bentuk dasarnya atau yang disebut dengan *stemming* untuk mengurangi perbedaan bentuk dari suatu kata, sehingga proses sistem temu kembali menjadi efektif (Yogatama, 2008).

2.2. *Stemming*

Stemming merupakan proses untuk mendapatkan *root/stem* atau kata dasar dari suatu kata dalam kalimat dengan cara memisahkan masing-masing kata dari kata dasar dan imbuhannya baik awalan (prefiks) maupun akhiran (sufiks). Sebagai contoh, kata bersama, kebersamaan, menyamai, akan di *stem* ke *root word* nya yaitu "sama".

Algoritma *stemming* untuk bahasa yang satu berbeda dengan algoritma *stemming* untuk bahasa lainnya. Sebagai contoh Bahasa Inggris memiliki morfologi yang berbeda dengan Bahasa Indonesia sehingga algoritma *stemming* untuk kedua bahasa tersebut juga berbeda. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks. Sedangkan pada teks berbahasa Indonesia lebih rumit/kompleks karena terdapat variasi imbuhan yang harus dibuang untuk mendapatkan *root word* dari sebuah kata.

Efektifitas algoritma *stemming* dapat dipengaruhi oleh beberapa factor (Mandala, Koryanti, Munir, & Harlili, 2004)

a. Kesalahan dalam proses pemenggalan imbuhan dari kata dasarnya. Kesalahan ini dapat berupa: *Overstemming*, *Understemming*, *Unchange* dan *Spelling exception*
b. Kekurangan dalam perumusan aturan penambahan imbuhan pada kata dasar.
c. Jumlah total aturan imbuhan yang didapat berhubungan dengan efektifitas proses temu kembali.

2.3. Algoritma Nazief & Adriani

Algoritma Nazief & Adriani dikembangkan pertama kali oleh Bobby Nazief dan Mirna Adriani. Algoritma ini berdasarkan pada aturan morfologi bahasa Indonesia yang luas, yang dikumpulkan menjadi satu grup dan di-enkapsulasi pada imbuhan/*affixes* yang diperbolehkan (*allowed affixes*) dan imbuhan/*affixes* yang tidak diperbolehkan (*disallowed affixes*). Algoritma ini menggunakan kamus kata dasar dan mendukung *recoding*, yakni penyusunan kembali kata-kata yang mengalami proses *stemming* berlebih.

Langkah-langkah algoritma Nazief & Adriani adalah:

1) Kata yang belum di-*stemming* dicari pada kamus, jika ditemukan, kata tersebut dianggap sebagai kata dasar yang benar dan algoritma dihentikan.
2) Hilangkan *Inflectional suffixes*, yaitu dengan menghilangkan *particle*("-lah", "-kah", "-tah" atau "-pun"), kemudian hilangkan *inflectional possessive pronoun suffixes* ("-ku", "-mu" atau "-nya"). Cek kata di dalam kamus kata dasar, jika ditemukan, algortima dihentikan, jika tidak lanjut ke langkah 3.
3) Hapus *Derivational Suffix* ("-i" atau "-an","). Jika kata ditemukan dalam kamus kata dasar, maka algoritma berhenti. Jika tidak, maka lanjut ke langkah 3a:
   a. Jika akhiran "-an" telah dihapus dan huruf terakhir dari kata tersebut adalah "-k", maka "-k" juga dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
   b. Akhiran yang dihapus ("-i", "- an" atau "-kan") dikembalikan, lanjut ke langkah 4.
4) Hapus *Derivational Prefix* ("be-","di-","ke-","me-","pe-","se-" dan "te-"). Jika kata yang didapat ditemukan didalam database kata dasar, maka proses dihentikan, jika tidak, maka lakukan *recoding*.

   Tahapan ini dihentikan jika memenuhi beberapa kondisi berikut:
   a. Terdapat kombinasi awalan dan akhiran yang tidak diijinkan
   b. Awalan yang dideteksi sama dengan awalan yang dihilangkan sebelumnya.
   c. Tiga awalan telah dihilangkan
5) Jika semua langkah telah dilakukan tetapi kata dasar tersebut tidak ditemukan pada kamus, maka algoritma ini mengembalikan kata yang asli sebelum dilakukan *stemming*.

Dalam penelitiannya Asian, dkk (Asian et al., 2005) melakukan beberapa pengembangan algoritma Nazief & Adriani sebagai berikut:

1) Menggunakan kamus kata yang lebih lengkap
2) Menambahkan aturan-aturan untuk kata-kata majemuk perulangan.
3) Menambahkan aturan awalan dan akhiran, serta aturan lainnya, yaitu:
   a. Menambahkan partikel (inflection suffix) "-pun".
   b. Penambahan aturan pemenggalan awalan.
   c. Perubahan aturan pemenggalan untuk tipe awalan "me".
4) Perubahan urutan proses *stemming*, yaitu:
   a. Kata dengan awalan "be-" dan akhiran "-lah", hilangkah awalan terlebih dahulu kemudian akhiran.
   b. Kata dengan awalan "be-" dan akhiran "-an", hilangkan awalan terlebih dahulu kemudian akhiran.
   c. Kata dengan awalan "me-" dan akhiran "-i", hilangkan awalan terlebih dahulu kemudian akhiran.
   d. Kata dengan awalan "di-" dan akhiran "-i", hilangkan awalan terlebih dahulu kemudian akhiran.
   e. Katadengan awalan "pe-" dan akhiran "-i", hilangkan awalan terlebih dahulu kemudian akhiran.
   f. Kata dengan awalan "ter-" dan akhiran "-i", hilangkan awalan terlebih dahulu kemudian akhiran.

2.4. Algortima Porter

Algoritma Porter adalah *stemmer* penggabungan yang diusulkan oleh Porter. Algoritma ini terdiri dari lima tahap, yang mensimulasikan proses infleksi dan derivasi dari kata. Pada setiap langkah, akhiran tertentu dihapus dengan cara substitusi. Aturan substitusi diterapkan ketika suatu kondisi terpenuhi.

2.5. Algoritma Porter untuk Bahasa Indonesia

Algoritma Porter untuk Bahasa Indonesia dikembangkan oleh Fadillah Z Tala (Tala, 2003) yang mengadopsi Algortima Porter untuk Bahasa Inggris yang di kembangkan oleh W.B Frakes. *Stemmer* ini menggunakan *rule base* analisis untuk mencari root sebuah kata. *Stemmer* ini sama sekali tidak menggunakan kamus sebagai acuan seperti *stemmer* Nazief Adriani, Ahmad, Vega atau Jelita. Langkah-langkah pada algoritma Porter adalah:

1. Menghapus partikel ("lah", "kah", "tah", "pun").
2. Mengapus kata ganti (*Possesive Pronoun*), seperti –ku, -mu, -nya
3. Mengapus awalan pertama. Jika tidak ditemukan, maka lanjut ke langkah 4a, dan jika ada maka lanjut ke langkah 4b.
4. a. Menghapus awalan kedua, dan dilanjutkan pada langkah ke 5a.
   b. Menghapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai kata dasar (*root word*). Jika ditemukan maka lanjut ke langkah 5b.
5. a. Menghapus akhiran dan kata akhir diasumsikan sebagai kata dasar (root word).
   b. Menghapus awalan kedua dan kata akhir diasumsikan sebagai kata dasar (*root word*). Dalam sebuah kata, memungkinkan adanya dua awalan yang saling berurutan.

Tidak semua awalan bisa ditambahkan pada awalan lainnya dalam sebuah kata. Aturan urutan awalan yang diperbolehkan seperti ditunjukkan pada Tabel 1.

Tabel 1. Urutan awalan yang diperbolehkan

Awalan 1	**Awalan 2**
Meng	Per
Di	Ber
Ter	
Ke	

III. METODE PENELITIAN

3.1 Pengumpulan data

Penelitian ini pada dasarnya lebih mengedepankan aspek implementasi dan analisis suatu teori, tentang *stemming* dengan menggunakan pendekatan algoritma Nazief Adriani dan Porter. Seperti pada penelitian-penelitian tentang analisis atau pembuktian suatu teori yang banyak menerapkan metode studi kepustakaan, demikian hal nya pada penelitian ini, penulis lebih banyak melakukan studi kepustakaan dengan mengambil referensi dari internet, ebook, paper, jurnal, skripsi, tesis, maupun buku-buku yang berkaitan dengan algoritma pemrograman, *stemming*, khususnya pada bahasa Indonesia, pembentukan kata pada Bahasa Indonesia, dan berbagai literatur ilmiah lainnya.

Referensi utama lain adalah penelitian yang dilakukan oleh Jelita Asian (Asian, 2007) dalam tesisnya yang berjudul "*Effective Techniques for Indonesian Text Retrieval* ", yang menerapkan algoritma Nazief Adriani, dan juga penelitian yang dilakukan oleh Fadillah Z Tala (Tala, 2003) dalam penelitiannya dengan judul "*A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*", yang meneliti tentang implementasi algoritma Porter untuk bahasa Indonesia.

Dalam penelitian ini, penulis menggunakan kamus kata dasar yang diunduh dari internet dengan jumlah kata 28.532 kata. Kata dasar tersebut kemudian disimpan dalam *database* yang nantinya akan digunakan dalam proses *stemming.* Sekumpulan tanda baca dan angka (48 karakter) disimpan dalam bentuk *file* yang akan dijadikan acuan untuk menghilangkan tanda baca dan angka pada tahap *preprocessing.* Tahap selanjutnya adalah penghapusan *stopword list. Stopword list* merupakan kata umum yang dianggap tidak memberikan informasi penting, sehingga keberadaannya bisa diabaikan, misalnya kata : yang, dan, yaitu, adalah dan lain sebagainya. Dalam penelitian ini penulis menggunakan

stopword list Tala yang dimuat pada penelitian Asian (Asian, 2007) yang terdiri dari 758 kata. Seperti halnya tanda baca, *stopword list* disimpan dalam bentuk *file* teks yang nantinya digunakan sebagai acuan untuk menghilangkan kata-kata yang termasuk dalam *stopwords.*

Penelitian dilakukan pada 26 dokumen uji yang diperoleh dari situs berita online (detik.com) pada tanggal 27 September 2015. Dokumen tersebut diunduh menggunakan *web crawler*, yang dibuat menggunakan *library* php bernama *PHP simple HTML DOM parser*, kemudian menyimpannya sebagai *file* XML (.xml), yang memiliki struktur sebagai berikut:

```
<document>
   <title>Judul Berita</title>
   <content>Isi Berita</content>
</document>
```

Untuk memudahkan pemanggilan dokumen uji ketika proses *stemming*, indeks dokumen tersebut disimpan dalam *database*.

3.2. Analisis sistem

Proses *stemming* diawali dengan mengubah semua huruf kapital menjadi huruf kecil (*lower case*), kemudian dilanjutkan dengan menghilangkan tanda baca dan angka dengan mengacu *file* tanda baca dan angka yang dibuat. Langkah selanjutnya adalah mengubah serangkaian kata tersebut menjadi kumpulan term (*corpus*), dilanjutkan dengan mengecek apakah ditemukan kata yang termasuk dalam *stopwords*. Jika ditemukan, kata tersebut akan otomatis dihilangkan. Demikian juga jika ditemukan kata yang muncul lebih dari satu kali (*double words*), maka kata tersebut juga akan dihilangkan dan hanya satu kata yang akan di proses.

3.3. Rancangan aplikasi

Pada penelitian ini dibangun sebuah aplikasi berbasis web dengan menggunakan Bahasa pemrograman PHP dan database MySQL untuk memudahkan dalam implementasi dan analisis algoritma *stemming* Nazief & Adriani dan Porter.

Aplikasi ini terdiri dari 6 halaman utama, yaitu Dokumen, *Stemming*, *Crawler*, Hasil, Kata Dasar, *Stopword*.

3.4. Analisis Hasil

Hasil *stemming* diperiksa secara manual untuk mengidentifikasi kata dasar yang benar atau kata yang mengalami kesalahan *stemming*. Begitu juga akan dicatat waktu proses *stemming* tersebut. Hasil akhir akan dihitung tingkat akurasinya dengan persamaan (1):

$$Akurasi = \left(\frac{RW}{W}\right) x100\% \quad (1)$$

dimana W adalah jumlah kata yang ter-*stem* dan RW adalah jumlah kata yang di-*stemming* dengan benar. Akurasi dinyatakan dalam persen (%).

IV. HASIL DAN PEMBAHASAN

4.1. Diagram Alir (*Flowchart*)

Pada penelitian ini digunakan 26 dokumen uji yang diunduh dari situs berita online (detik.com) pada tanggal 27 September 2015 menggunakan *crawler* Simple HTML Dom. Data yang diunduh disimpan dalam database untuk indeks dan judulnya, sedangkan kontennya disimpan dalam bentuk file XML.

Seperti diilustrasikan pada Gambar 1, proses *stemming* diawali dengan menghilangkan tanda baca dan angka pada dokumen uji, kemudian di *split* menghasilkan

corpus. *Corpus* ini dicek setiap katanya apakah termasuk dalam *stopwords* atau tidak. Dicek juga apakah kata-kata tersebut muncul lebih dari satu kali (*double words*). Kata yang termasuk *stopwords* dan *double words* ini dihilangkan dan menghasilkan *corpus* baru. *Corpus* baru ini yang kemudian dilakukan proses *stemming* menggunakan algoritma Nazief & Adriani dan Porter. *Root word* yang didapat disimpan di database, kemudian dilakukan analisis dan dibandingkan hasilnya.

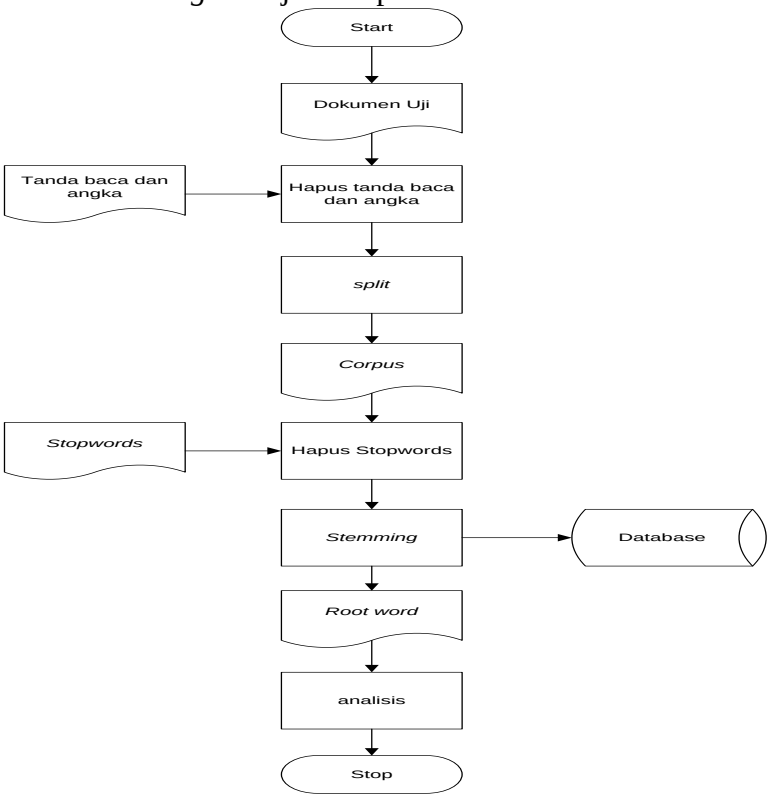
4.2. Implementasi

Proses pengujian dilakukan dengan menerapkan algoritma Nazief & Adriani dan algoritma Porter untuk mencari akar kata. Proses ini diawali dengan mengubah semua huruf kapital menjadi huruf kecil (*lower case*), kemudian dilanjutkan dengan menghilangkan kata ganda (*double word*), yang artinya untuk kata yang sama, *stemming* akan dilakukan hanya pada satu kata. Untuk kata yang termasuk dalam *stopword* juga akan dieliminir dan tidak akan dilakukan proses *stemming*.

Pada penelitian ini terdapat 8.168 kata yang diujikan, 1.312 diantaranya termasuk kata yang muncul lebih dari satu kali (*double word*), 4.724 diantara termasuk dalam *stopword* dan 2.132 kata yang benar-benar diujikan pada kedua *stemmer* yang diterapkan pada penelitian ini.

Pengujian hasil *stemming* dilakukan secara manual untuk menentukan hasil yang didapat merupakan kata dasar yang benar atau salah karena mengalami kesalahan *stemming*.

Pengujian dengan menerapkan algoritma Porter, dimana proses *stripping* dilakukan tanpa menggunakan kamus kata dasar.Dari 2.132 kata yang di *stem* menghasilkan 1.687 kata dasar yang benar dan 445 kata (21%) yang mengalami kesalahan *stemming*, terdiri dari 67*understemming*, 374 *overstemming* dan 4 *unchange*. Waktu proses untuk algoritma Porter di bawah 1 detik, dengan total waktu proses $\pm$12 detik. Beberapa contoh kata yang mengalami kesalahan *stemming* ditunjukkan pada Tabel 2.



Gambar 1. *Flowchart* proses *stemming*

Tabel 2. Contoh kesalahan *stemmer* Porter

Word	**Stem**	**Kesalahan**
Korban	korb	*Overstemming*
Kerja	rja	*Overstemming*
Secepatnya	secepat	*understemming*
Tawarin	tawarin	*Unchange*

Pengujian dengan menerapkan algoritma Nazief & Adriani (NA) berbeda dengan Porter, dimana algortima ini menggunakan kamus kata dasar sebagai acuan dalam proses *stripping*. Dari 2.132 kata yang di *stem* menghasilkan 2.031 kata dasar yang benar dan 101 kata (5%) yang mengalami kesalahan *stemming*, terdiri dari 23 *understemming*, 27 *overstemming*, 51 *unchange*. Total waktu proses untuk algoritma Nazief & Adriani $\pm$22 detik. Beberapa contoh kata yang mengalami kesalahan *stemming* diperlihatkan pada Tabel 3.

Tabel 3. Contoh kesalahan *stemmer* Nazief & Adriani

Word	**Stem**	**Kesalahan**
Dinilainya	nila	*Overstemming*
Beragam	agam	*Overstemming*
Memiliki	memilik	*Understemming*
Tindakan	tindakan	*Unchange*
Peringkat	peringkat	*Unchange*
Mengecek	ecek	*Understemming*
Tawarin	tawarin	*Unchange*

4.3. Analisis Hasil

Dari total 2.132 kata yang diuji, algortima Nazief & Adriani menghasilkan root word yang benar sebanyak 2.031 kata, sedangkan algoritma Porter menghasilkan root word yang benar sebanyak 1.687 kata. Tingkat kesalahan *stemmer* Nazief & Adriani hanya 5%, sangat jauh jika dibandingkan dengan Porter yaitu 21%. Tetapi *stemmer* Nazief &Adriani membutuhkan waktu proses hampir 2 kali lebih lama jika dibanding algoritma Porter.

Berdasarkan hasil yang didapat pada kedua *stemmer* yang diujikan, dapat dihitung tingkat akurasi dari kedua *stemmer* seperti ditunjukkan pada Tabel 4.

Table 4. Perbandingan hasil *stemming* Porter dan Nazief & Adriani

Algoritma	**Akurasi**	**Waktu proses**
Porter	79,13%	12.3822753429
Nazief & Adriani	95,26%	22.1668348312

Dari hasil tersebut tampak, bahwa algoritma Nazief & Adriani memberikan hasil yang lebih baik untuk *stemming* pada dokumen berbahasa Indonesia. Namun untuk efisiensi waktu proses, algoritma Porter lebih baik dibanding algoritma Nazief & Adriani.

V. KESIMPULAN DAN SARAN

5.1. Kesimpulan

1. Beberapa kesalahan *stemming* pada algortima Porter, terjadi karena kurangnya aturan *stripping* untuk kata-kata yang mengalami peleburan huruf, contoh: menuai, mengirim, dan lain sebagainya.
2. Kesalahan *overstemming* pada algortima Porter terjadi karena kesalahan sistem mengenali imbuhan yang ada.
3. Beberapa kesalahan *stemming* pada algortima Nazief & Adrini terjadi karena kamus kata yang tidak lengkap, kurangnya aturan *stripping* serta kesalahan urutan algortima.
4. Algoritma Porter dan Nazief & Adriani belum bisa menangani untuk kata tidak baku atau berakhian "-in".

5. Algortima Porter dan Nazief & Adriani belum bisa menangani untuk kata-kata berakhiran “wan”,”man”,“wati” dan ”isasi”
6. Algoritma Porter dan Nazief & Adriani tidak bisa menangani untuk kata-kata yang bukan termasuk Bahasa Indonesia (Bahasa Inggris), contoh: mendownload, mengupload.
7. Algoritma Porter dan Nazief & Adriani mengalami ambiguitas untuk menentukan kata dasar dari kata “menggulai”.
8. Algoritma Nazief & Adriani memberikan hasil yang lebih baik di banding Porter untuk mendukung sistem temu kembali informasi.
9. Algoritma Porter memiliki waktu proses yang lebih cepat jika dibanding dengan algortima Nazief & Adriani.

5.2. Saran

1. Untuk mengurangi tingkat kesalahan *stemming*, gunakan kamus kata dasar yang lebih lengkap.
2. Pada algoritma Nazief & Adriani, untuk awalan "men" dan akhiran "i", hapus awalan terlebih dahulu, tambahkan huruf "t" diawal. Jika huruf kedua dari belakang adalah konsonan, maka hilangkan akhiran "i", jika vocal, maka biarkan akhiran "i".
3. Tambahkan aturan *stripping* untuk akhiran “wan”,”man”,”wati” dan “isasi” pada kedua *stemmer*.

UCAPAN TERIMA KASIH

Penulis mengucapkan banyak terima kasih kepada para reviewer yang telah memberikan koreksi dan masukan untuk perbaikan sampai terpublikasinya naskah ini.

DAFTAR PUSTAKA

Afuan, L. (2013). Stemming Dokumen Teks Bahasa Indonesia. *Jurnal Telematika*, Vol. 6, No. 2, Hal. 34–40.

Asian, J. (2007). *Effective Techniques for Indonesian Text Retrieval*. *Thesis of Doctoral*. RMIT University.

Asian, J., Williams, H. E., & Tahaghoghi, S. M. M. (2005). Stemming Indonesian. In *Conferences in Research and Practice in Information Technology Series*, Vol. 38, Hal. 307–314. http://doi.org/10.1145/1316457.1316459

Budhi, G. S., Gunawan, I., & Yuwono, F. (n.d.). Algoritma Porter Stemmer For Bahasa Indonesia Untuk Pre-Processing Text Mining Berbasis Metode Market Basket Analysis. *Paper No. 63 UTY, Universitas Kristen Petra*, (264017), 3–4.

Mandala, R., Koryanti, E., Munir, R., & Harlili. (2004). Sistem Stemming Otomatis untuk Kata dalam Bahasa Indonesia. In *Seminar Aplikasi Teknologi Informasi*, pp. 29–36. Yogyakarta.

Mario Naga Mait, H. (2012). *Making Stemming Synonym Indonesian Using Algorithm Nazief And Adriani*. Gunadarma University.

Marsya, J. M., & Abidin, T. F. (2011). Analisa dan Evaluasi Afiks Stemming untuk Bahasa Indonesia. *Seminar Nasional Dan Expo Teknik Elektro*, pp. 46–51.

Tala, F. Z. (2003). A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia. In *Institute for Logic, Language and Computation Universeit Van Amsterdam*.

Utomo, M. S. (2013). Implementasi Stemmer Tala pada Aplikasi Berbasis Web. *Jurnal Teknologi Informasi DINAMIK*, Vol .18, No. 1, 41–45.

Yogatama, D. (2008). *Studi Penggunaan Stemming untuk Meningkatkan Performansi Sistem Temu Balik Informasi*. Departemen Teknik Informatika ITB.