Jurnal Coding, Sistem Komputer Untan
Volume 06, No.02 (2018), hal. 13-24 ISSN: 2338-493X

IMPLEMENTASI METODE *NAIVE BAYES CLASSIFIER* UNTUK APLIKASI *FILTERING EMAIL SPAM* DENGAN *LEMMATIZATION* BERBASIS *WEB*

Harry Pribadi Fitrian [1], Ikhwan Ruslianto[2], Rahmi Hidayati[3]
[1,2,3]Jurusan Sistem Komputer, Fakultas MIPA Universitas Tanjungpura
Jalan Prof Dr. H. Hadari Nawawi Pontianak
Telp./Fax. : (0561) 577963
e-mail: [1] lubisharry7@gmail.com, [2]ikhwanruslianto@siskom.untan.ac.id,
[3]rahmihidayati@siskom.untan.ac.id

Abstrak

Surat elektronik (email) merupakan salah satu media komunikasi dalam jaringan internet yang paling populer. Surat elektronik (email) menyediakan layanan yang cepat, gratis, serta memiliki kemampuan dapat mengirim pesan ke banyak penerima dalam waktu singkat. Terdapat pihak tertentu yang menyalahgunakannya dengan mengirim email berisi promosi produk atau jasa, pornografi, virus, dan hal-hal yang tidak penting ke pengguna email lainnya. Email tersebut disebut email SPAM (Stupid Pointless Annoying Message). Email SPAM mengakibatkan adanya kelainan pada penggunaan traffic jaringan, storage space, dan computational power. Penelitian ini bertujuan untuk membuat suatu aplikasi filtering email yang memanfaatkan metode naive bayes classifier untuk mengklasifikasikan jenis email termasuk email SPAM atau HAM dan lemmatization untuk mengolah kata menjadi kata dasar. Berdasarkan hasil pengujian menggunakan 131 sampel email, berhasil mengklasifikasikan dengan benar sebanyak 119 file. Sedangkan 12 file yang diuji mendapatkan nilai prediksi yang salah. Nilai akurasi yang didapatkan pada penelitian ini adalah 90,83%.

Kata Kunci : *email, SPAM, HAM, lemmatization, Naive Bayes Classifier.*

1. PENDAHULUAN

Surat elektronik (*email*) merupakan salah satu media komunikasi dalam jaringan internet yang paling populer. Proses yang dapat dilakukan dalam *email* adalah berdiskusi (*discussion*), transfer informasi berupa *file* (*mail attachment*) dan mempromosikan dan menyebarkan informasi seperti produk iklan. Penggunaan *email* di Indonesia terbilang cukup besar. Indonesia menduduki peringkat ketiga dalam penggunaan *email* dengan persentase 38,5%[1].

Email menyediakan layanan yang cepat, gratis, dan memiliki kemampuan dapat mengirim pesan ke banyak penerima dalam waktu singkat. Keuntungan email tersebut disalahgunakan oleh beberapa pihak dengan mengirimkan *email* berisi promosi produk atau jasa, pornografi, virus, dan hal-hal yang tidak penting ke pengguna *email* lainnya. *Email* tersebut dapat disebut *email SPAM* (*Stupid Pointless Annoying Message*). Pada survey ditemukan bahwa 10% dari *mail* yang diterima oleh suatu perusahaan adalah *SPAM-mail*[2].

Para penyedia layanan *email* seperti *Google*, *Yahoo,* dan *Outlook* telah mempersiapkan layanan untuk menyaring suatu *email*. Namun, tidak sedikit ditemukan kesalahan seperti *email* yang seharusnya asli ternyata dianggap *SPAM* oleh penyedia layanan atau sebaliknya.

Penelitian yang dilakukan oleh Indrastanti R. Widiasari pada tahun 2013 dalam jurnal yang berjudul Pembangunan *SPAM E-Mail Filtering System* dengan Metode *Naive Bayesian*. Dalam jurnal tersebut, Widiasari melakukan penyaringan *email* dengan melihat nilai *threshold value* yang masuk, jika lebih dari 50% maka akan masuk ke *email* yang sah. Jika kurang dari 50%, maka akan terkategori sebagai *email SPAM*. Berdasarkan 150 *email* yang masuk melalui *email client*, aplikasi buatannya mempunyai akurasi sebesar 96,67%.[3]

Persentase keberhasilan dalam *filtering email* dengan menggunakan metode *naive*

Bayesian sangat tinggi seperti penelitian yang telah dilakukan oleh Andrean Taufik pada tahun 2017 dalam jurnal berjudul Analisis *Filtering Email SPAM* dengan Menggunakan Metode *Naive Bayesian.* Dalam jurnalnya tersebut, yang diteliti dari total 867 *file*, berhasil mengklasifikasikan dengan benar sebanyak 850 *file*. Sedangkan 17 *file* yang diuji mendapatkan nilai prediksi yang salah. Sehingga akurasi yang didapatkan pada penelitian ini adalah 98,04%.[1]

Penelitian yang dilakukan oleh Tom Christiansen pada tahun 2012 dalam jurnal berjudul *BioLemmatizer: a lemmatization tool for morphological processing of biomedical text.* Dalam jurnalnya tersebut, dapat ditarik kesimpulan yaitu *lemmatization* dapat digunakan untuk mengenali domain biologis berdasarkan penggunaan kata *leksikon*, dan mendefinisikan seperangkat aturan yang mengubah kata menjadi *lemma* jika tidak ditemui dalam *leksikon*. *Leksikon* adalah kamus biological. *BioLemmatizer* mencapai akurasi 97,5% dalam mengenali satu set evaluasi disiapkan dari corpus CRAFT(koleksi artikel biomedis teks lengkap) dan akurasi 97,6% pada korpus LLL0.[4]

Adapun pada penelitian ini akan dibahas mengenai "Implementasi Metode *Naive Bayes Classifier* Untuk Aplikasi *Filtering Email SPAM* dengan *Lemmatization* Berbasis *Web*". Untuk mengenali teks berbahasa Indonesia, dilakukan proses *lemmatization*, sedangkan untuk mengklasifikasikan *email*, dilakukan proses klasifikasi *naive bayes classifier*. Kemudian hasil dari proses *lemmatization* dilakukan *pre-processing* untuk melanjutkan tahap selanjutnya yaitu proses klasifikasi *naive bayes.* Hasil dari sistem berupa informasi mengenai akurasi penggunaan *naive bayes classifier* pada aplikasi *filtering email SPAM*.

2. LANDASAN TEORI

2.1 *Email*

Email adalah sebuah layanan komunikasi yang dapat mengirim surat dalam bentuk elektronik dan berbasis internet. Email terhubung dalam jaringan, sehingga pengguna dapat saling mengirim pesan ke pengguna lain. Sistem email dapat dibandingkan dengan rumit dan luasnya jaringan telepon, namun lebih unggul dibanding segi teknis dan sistem pemrosesan data[5].

Email memiliki 3 komponen. yaitu, envelope yang unit yang pertama kali dikirimkan. *Header* untuk informasi detail email, yang terakhir adalah *content / body* yaitu isi pesan yang akan dikirim atau diterima

2.2 *SPAM*

SPAM adalah pesan yang dikirim secara asal dan acak ke individu maupun kelompok. *SPAM* seringkali berisi konten ataupun lampiran yang berbahaya. Penyalahgunaan ini disebabkan oleh karena penggunaan internet yang murah dan mudah sehingga *SPAM* menjadi sangat mudah disebarluaskan.

Terdapat 3 tujuan penyebaran *SPAM*[6]:

1. Menarik perhatian agar situs atau produk dapat dilihat.
2. Memasarkan produk kepada penerima.
3. Memasang *malware*.

SPAM menurunkan produktivitas penerima, sehingga pengembangan alat penyaring *email* yang efektif dan efisien sangat penting. Berikut beberapa alasan mengapa deteksi *SPAM* sangat penting:

1. Banyak *SPAM* yang berbahaya bagi komputer korban.
2. *SPAM* menghabiskan waktu dan produktivitas korban, karena tidak ada informasi berharga dari *SPAM*.
3. *SPAM* dapat menghabiskan sumber daya

2.3 Text Mining

Text mining adalah bidang ilmu yang mengacu kepada pencarian informasi, *data mining*. *Text mining* adalah proses mencari dan mengesktrak informasi yang berguna dari data berbentuk teks. *Text mining* bertujuan untuk mendapatkan infomasi dari teks yang tidak mempunyai struktur[7].

Proses *Text Mining* mencakup *preprocessing*, pemodelan, dan evaluasi. *Preprocessing* dilakukan untuk mengubah teks dari bentuk yang belum terstruktur menjadi bentuk yang terstruktur. Pada penelitian ini teknik pemodelan yang digunakan adalah klasifikasi dengan algoritma *naive bayesian*. Sedangkan untuk evaluasi hasil klasifikasi adalah dengan menggunakan *confusion matrix*. *Confusion matrix* adalah tabel yang menyatakan jumlah data uji yang diklasifikasikan benar dan yang diklasifikasikan salah[8].

Tabel 1. *Confusion Matrix*

Correct Classification	*Classified as*	
	+	-
+	*True Positives*	*False Negatives*
-	*False Positives*	*True Negatives*

Pengertian nilai pada tabel diatas adalah:
1. *True Positives* adalah jumlah *record* data positif yang diklasifikasikan sebagai nilai positif.
2. *False Positives* adalah jumlah *record* data negatif yang diklasifikasikan sebagai nilai positif.
3. *False Negatives* adalah jumlah *record* data positif yang diklasifikasikan sebagai nilai positif.
4. *True Negatives* adalah jumlah *record* data negatif yang diklasifikasikan sebagai nilai negatif.

Nilai yang dihasilkan *Confusion matrix* adalah:
1. *Accuracy*, persentase jumlah *record* data yang diklasifikasikan secara benar oleh algoritma.
2. *Precision / Confidence*, proporsisi jumlah kasus yang diprediksi positif dan data sebenarnya juga bernilai positif.
3. *Recall / Sensitivity*, proporsisi jumlah kasus positif yang sebenarnya diprediksi positif secara benar.

2.4 *Naive Bayesian*

Algoritma *naive bayesian* atau *naive bayes classifier* merupakan metode yang digunakan untuk memprediksi suatu kejadian pada masa yang akan datang. Penggunaan probabilitas kata dijadikan sebagai masukan probabilitas dari kejadian. Klasifikasi *naive bayesian* akan melihat data lama dalam menentukan kemiripan dengan data yang baru. Sehingga dalam klasifikasi harus ada data lama sebagai pembanding[9].

Algoritma *naive bayesian* mempunyai asumsi *class-conditional independence* yaitu setiap atribut tidak memiliki keterkaitan dengan atribut lain. Tujuan asumsi ini untuk mengurangi proses komputasi. Algoritma *naive bayesian* merupakan pengembangan dari teori *bayes*:

$$P(H|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|H)P(H)}{P(\boldsymbol{X})} \quad (1)$$

Pada teori *Bayes* diatas, diimplementasikan ke dalam algoritma *naive bayesian*. Dalam algoritma ini, setiap dokumen direpresentasikan dengan atribut $x_1, x_2, x_3, \dots x_n$. Algoritma *naive bayesian* membandingkan probabilitas setiap kata dalam dokumen *X* terhadap semua kategori yang ada.

$$P(C_i|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|C_i)P(C_i)}{P(\boldsymbol{X})} \quad (2)$$

Karena nilai dokumen $P(\boldsymbol{X}) = P(x_1, x_2, x_3, \dots x_n)$ konstan untuk semua kategori, sehingga pada persamaan diatas dapat diuraikan menjadi:

$$P(C_i|\boldsymbol{X}) = P(\boldsymbol{X}|C_i)P(C_i) \quad (3)$$

$$P(\boldsymbol{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) \quad (4)$$

$$P(\boldsymbol{X}|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \dots \times P(x_n|C_i) \quad (5)$$

Setelah mendapatkan nilai probabilitas setiap dokumen terhadap masing-masing kategori (m adalah jumlah kategori), bandingkan dan cari nilai probabilitas tertinggi dari semua dokumen yang diuji. Jika dan hanya jika:

$$P(\boldsymbol{X}|C_i)P(C_i) > P(\boldsymbol{X}|C_j)P(C_j) \; untuk \; 1 \leq j \leq m, j \neq 1 \quad (6)$$

Untuk $P(C_m)$ dan $P(x_i|C_m)$ dihitung pada saat pelatihan, dimana persamaannya adalah sebagai berikut:

$$P(C_m) = \frac{|docs\ m|}{|docs|} \quad (7)$$

$$P(x_i|C_m) = \frac{n_k+1}{n+|words|+2} \quad (8)$$

Keterangan:
1. $|docs\ m|$ adalah jumlah semua dokumen pada kategori m.
2. $|docs|$ adalah jumlah semua dokumen pada data latih.
3. n_k adalah jumlah frekuensi kemunculan setiap kata x_i pada kategori C_m.

4. n adalah jumlah frekuensi kemunculan kata dari setiap kategori.
5. $|words|$ adalah jumlah kata dari semua kategori yang ada pada data latih.

2.5 *Lemmatization*

Lemmatization adalah transformasi morfologis untuk mengubah kata yang muncul dalam teks ke dalam bentuk dasar atau kamus yang dikenal sebagai *lemma*. Hal ini dilakukan dengan cara menghapus akhiran dan awalan kata. *Lemma* sesuai dengan bentuk tunggal dalam kasus kata benda, bentuk infinitif dalam kasus kata kerja, dan bentuk positif dalam kasus kata sifat atau kata keterangan. *Lemmatization* adalah proses normalisasi di mana varian morfologi yang berbeda dari sebuah kata dipetakan ke *lemma* yang sama sehingga bisa dianalisis sebagai satu *item* (istilah atau konsep).

Dengan mengurangi jumlah istilah yang berbeda, *lemmatization* menurunkan kompleksitas teks yang dianalisis, dan oleh karena itu membawa manfaat penting bagi komponen pengolahan teks. Ketika digabungkan dalam sistem pencarian informasi, *lemmatization* dapat membantu untuk meningkatkan pencarian keseluruhan karena *queri* akan dapat mencocokkan lebih banyak dokumen saat varian dalam *queri* dan dokumen dinormalisasi secara morfologis.

Stemming, kata lain berupa teknik normalisasi, sudah banyak diterapkan dalam pencarian informasi. *Stemming* menormalkan beberapa varian morfologi dari sebuah kata ke dalam bentuk yang sama, yang dikenal sebagai *stem*. Hal ini dilakukan dengan cara menanggalkan akhiran sebuah kata. Meskipun tujuan *stemming* serupa dengan *lemmatization*, perbedaan penting adalah bahwa *stemming* tidak bertujuan untuk menghasilkan bentuk kamus kata yang terjadi secara alami. *Stem* dari kata "*regulated*" akan menjadi "*regul*" daripada bentuk kata kerja dasar yang sebenarnya "*regulate*". Hal ini sering mengakibatkan penggabungan istilah semantis yang salah. Adapun contoh lain dari istilah "*activates*", "*activations*" dan "*activities*" semuanya akan berakhir pada "*activ*" atau "*act*" oleh kebanyakan *algoritma stemming*, sementara *lemmatizer* akan membuat kata tersebut memiliki bentuk dasar yang berbeda (perhatikan bahwa "*activates*" dan "*activation*" akan diartikan berbeda oleh *lemmatizer* yang hanya menangani morfologi infleksi karena yang pertama adalah bentuk kata kerja dan yang terakhir adalah kata benda). Di sisi lain, *algoritma stemming* yang ada mungkin tidak benar menguraikan bentuk infleksi terkait, seperti "*actor*" dan "*action*" (*understemming*). Hal ini membuktikan bahwa *lemmatization* lebih baik daripada *stemming* dalam kasus membentuk suatu kata yang telah terurai[5].

3. METODE PENELITIAN

Proses pertama yang akan dimulai dengan studi literatur yang terkait dengan metode *naive bayes classifier*, *email*, dan *lemmatization*. Selanjutnya dilakukan pengumpulan data dan dilanjutkan dengan analisis kebutuhan untuk penelitian ini. Setelah data terkumpul, lalu dibuatlah perancangan sistem yang kemudian diintegrasikan menjadi suatu sistem sehingga berfungsi sebagaimana mestinya. Selanjutnya dilakukan pengujian dan implementasi untuk mengetahui kinerja sistem apakah sistem yang dibuat berkerja sesuai dengan tujuan penelitian. Setelah itu membuat kesimpulan dan saran dari proses penelitian. Diagram alir penelitian dapat dilihat pada Gambar 1.

Gambar 1. Diagram Alir Penelitian

4. PERANCANGAN SISTEM

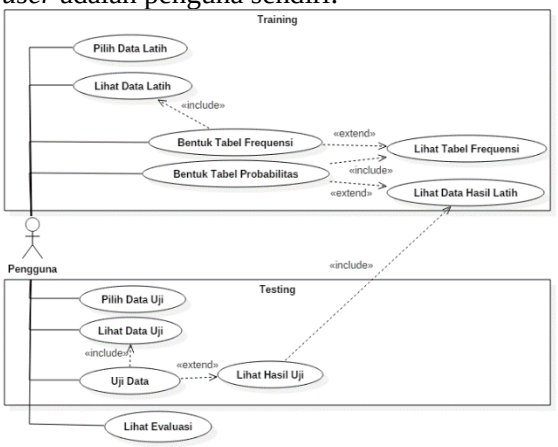
4.1 Cara Kerja Aplikasi

Terbagi atas dua bagian utama, yaitu pelatihan dan pengujian. Kedua bagian tersebut sama-sama melewati *preprocessing*. Untuk pelatihan, kata akan disimpan dan dibentuk tabel frekuensi. Setelah tabel frekuensi dibentuk, algoritma *naive bayesian* diimplementasikan dan akan menghasilkan nilai probabilitas tiap kategori untuk semua kata.Sedangkan untuk pengujian, semua kata

yang ada pada *file* yang sedang diuji akan dibandingkan dengan daftar kata pada tabel probabilitas. Setelah mendapat nilai probabilitas untuk *file* yang diuji maka nilai prediksi akan diperbarui.

4.2 Rancangan Aplikasi

Use Case dibuat untuk mendeskripsikan peran *user* dalam sistem. Pada Penelitian ini *user* adalah penguna sendiri:

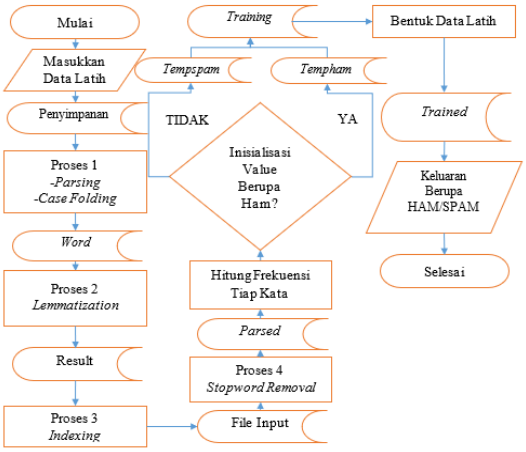


Gambar 2. *Use Case Diagram*

4.3 Proses Pelatihan

Secara garis besar, alur pelatihan data dibagi menjadi 3 bagian utama. *Import* untuk memasukkan *file* dan mengubahnya ke bentuk terstruktur dengan *preprocessing*. Setelah *import*, proses *training* dilakukan. Menghitung frekuensi kemunculan setiap kata memerlukan dua tabel *temporary* yaitu "tempspam" dan "tempham" agar bisa membentuk data pada tabel "training". Bagian terakhir, yaitu proses bentuk data. Setelah frekuensi semua kata untuk setiap kategori telah ditentukan, hitung nilai probabilitas semua kata berdasarkan kategori, lalu simpan pada tabel "trained".

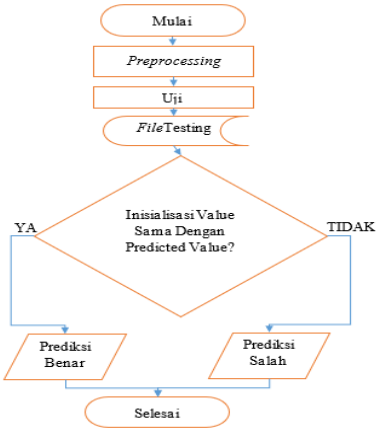
Proses pelatihan data digambarkan dengan diagram alir dibawah:



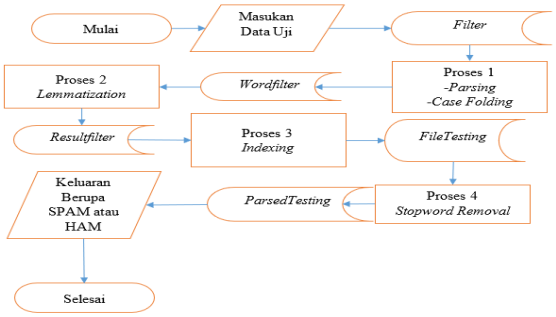
Gambar 3. Diagram Alir Pelatihan

4.4 Proses Pengujian

Proses pengujian hampir sama dengan proses pelatihan. Secara garis besar, alur pengujian data dibagi menjadi 3 bagian utama. *Import* untuk memasukkan *file* dan mengubahnya ke bentuk terstruktur dengan *preprocessing*. Setelah *import*, proses pengujian atau klasifikasi dilakukan. Membandingkan *file* yang diuji dengan data yang ada pada tabel "trained", mengubah nilai *predicted_value* dan menampilkan hasilnya. Proses pengujian data digambarkan dengan diagram alir pengujian utama dan diagram alir pengujian *preprocessing* dibawah:



Gambar 4. Diagram Alir Pengujian Utama



Gambar 5. Diagram Alir *Pengujian* Preprocessing

4.5 Contoh Pelatihan dan Pengujian

Berikut contoh data untuk pelatihan:

Tabel 2. Data Pelatihan

No	Nama *File*	Konten	Nilai Awal
1	Test.txt	Uang gratis ($ 500) pada iklan	*SPAM*
2	*File*1.txt	Cek *website* kami untuk referensi	HAM
3	*File*2.txt	Kunjungi *website* kami untuk gratis	HAM
4	*File*3.txt	Translate *website* ini gratis	HAM
5	Tests.txt	Ikut link ini atau surati kami untuk gratis	*SPAM*

Tabel diatas setelah melewati *preprocessing* maka akan menjadi:

Tabel 3. Data Setelah *Preprocessing*

Nama *File*	Konten	Nilai Awal
Test.txt	Gratis	*SPAM*
	Uang	*SPAM*
	Iklan	*SPAM*
*File*1.txt	Cek	HAM
	Website	HAM
	Referensi	HAM
*File*2.txt	Kunjung	HAM
	Website	HAM
	Gratis	HAM
*File*3.txt	Translate	HAM
	Website	HAM
	Inggris	HAM
Tests.txt	Ikut	*SPAM*
	Link	*SPAM*
	Surat	*SPAM*
	Gratis	*SPAM*
	Uang	*SPAM*

Proses selanjutnya adalah membentuk tabel frekuensi:

Tabel 4. Tabel Frequensi

No	Kata	Frekuensi SPAM	Frekuensi HAM
1	Gratis	2	1
2	Uang	2	0
3	Iklan	1	0
4	Cek	0	1
5	Website	0	3
6	Referensi	0	1
7	Kunjung	0	1
8	Translate	0	1
9	Inggris	0	1
10	Ikut	1	0
11	Link	1	0
12	Surat	1	0

Selanjutnya, menghitung probabilitas setiap kategori pada data latih:

- $P(SPAM) = \frac{2}{5} = 0{,}4$
- $P(HAM) = \frac{3}{5} = 0{,}6$

Pada contoh perhitungan ini, penulis akan menampilkan perhitungan untuk satu dokumen sebagai contoh, yaitu dokumen “Tests.txt”:

- $words = 12$
  jumlah semua kata untuk semua kategori
- $n\,SPAM = 8$
  jumlah frekuensi semua kata pada SPAM
- $n\,HAM = 9$
  jumlah frekuensi semua kata pada HAM

Tabel 5. Tabel Probabilitas

Kata	SPAM Prob.	HAM Prob.
Ikut	$P(ikut\|SPAM) = \frac{1+1}{10+12} = 0{,}091$	$P(ikut\|HAM) = \frac{0+1}{11+12} = 0{,}0435$
Link	$P(link\|SPAM) = \frac{1+1}{10+12} = \frac{2}{22} = 0{,}091$	$P(link\|HAM) = \frac{0+1}{11+12} = \frac{1}{23} = 0{,}0435$
Surat	$P(surat\|SPAM) = \frac{1+1}{10+12} = \frac{2}{22} = 0{,}091$	$P(surat\|HAM) = \frac{0+1}{11+12} = \frac{1}{23} = 0{,}0435$
Gratis	$P(gratis\|SPAM) = \frac{2+1}{10+12} = \frac{3}{22} = 0{,}136$	$P(gratis\|HAM) = \frac{1+1}{11+12} = \frac{2}{23} = 0{,}087$
Uang	$P(uang\|SPAM) = \frac{2+1}{10+12} = \frac{3}{22} = 0{,}136$	$P(uang\|HAM) = \frac{0+1}{11+12} = \frac{1}{23} = 0{,}0435$

Maka perhitungan probabilitas tiap kata untuk dokumen "Tests.txt" untuk setiap kategori ada pada tabel 5:

Setelah mendapatkan nilai pada tabel probabilitas maka pelatihan berakhir, karena sudah bisa diuji. Perhitungan klasifikasi dokumen X adalah dengan mengalikan semua nilai probabilitas pada dokumen yang sama $X = \{x_1, x_2, \dots, x_{n\}}$ dengan nilai probabilitas kategori $P(C_m)$. Dokumen yang dipilih adalah dkumen "tests.txt" yang diinisialisasikan SPAM. Sehingga untuk perhitungan dokumen "tests.txt" adalah sebagai berikut:

- $P(Tests.txt \mid HAM) = P(HAM) \times P(ikut|HAM) \times P\,P(surat|HAM) \times P(gratis|HAM) \times P(uang|HAM)$
- $P(Tests.txt \mid SPAM) = P(SPAM) \times P(ikut|SPAM) \times P(link|SPAM) \times P(surat|SPAM) \times P(gratis|SPAM) \times P(uang|SPAM)$

Maka nilai akhirnya adalah sebagai berikut:

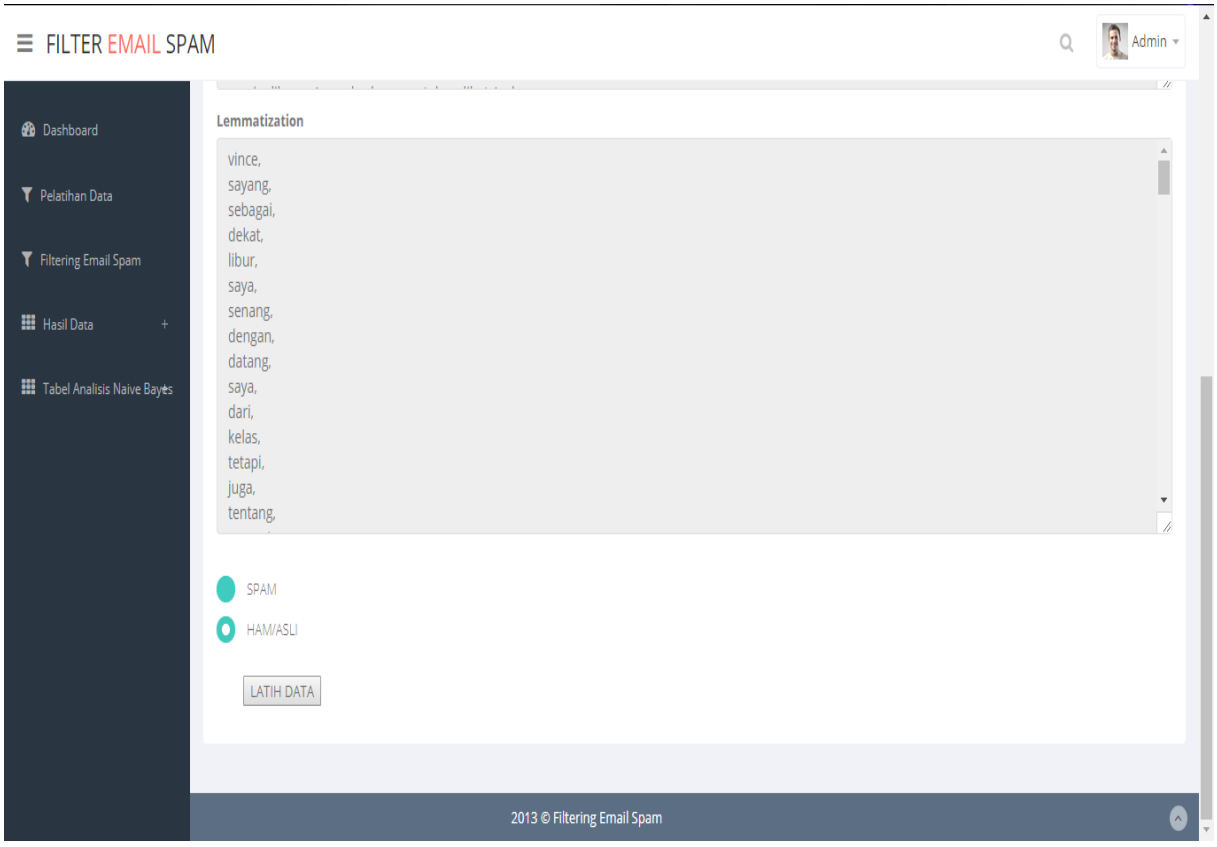
- $P(Tests.txt \mid HAM) = 0{,}00000019$
- $P(Tests.txt \mid SPAM) = 0{,}0000056$

Setelah melakukan proses perhitungan probabilitas dokumen, selanjutnya adalah membandingkan nilai dokumen yang sama untuk setiap kategori. Karena $P(Tests.txt \mid SPAM) > P(Tests.txt \mid HAM)$, maka nilai prediksi dokumen "tests.txt" adalah bernilai SPAM. Prediksi benar karena nilai awal dan nilai prediksi sama.

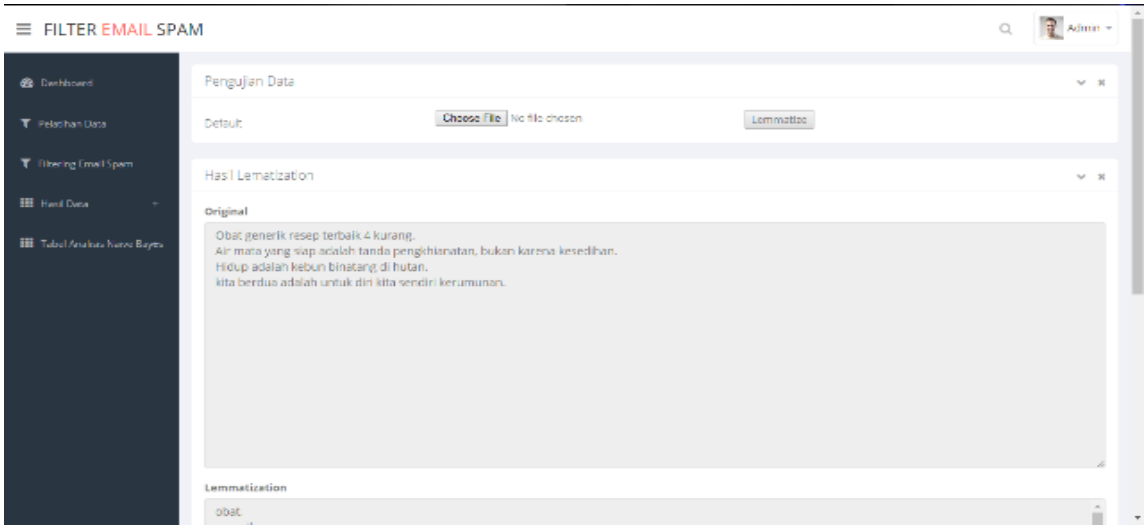
5. HASIL DAN PEMBAHASAN

5.1 Implementasi Sistem

Halaman pelatihan data pada gambar 10 dan 11 dibuat sesuai dengan perancangan sistem dengan memiliki tombol masukan untuk *file* dokumen dan tombol untuk proses *lemmatization* dimulai. Halaman ini juga memiliki *textarea* yang digunakan untuk menampilkan isi asli dan hasil dari *lemmatization* untuk dokumen tersebut. Halaman ini juga memiliki 2 tombol HAM dan SPAM untuk memberikan inisialisasi awal pada dokumen tersebut. Halaman ini juga memiliki tombol latih data untuk melanjutkan proses pelatihan.



Gambar 10. Halaman Pelatihan data



Gambar 11. Halaman Pengujian data

Halaman pengujian data pada gambar 10 dan 11 hampir sama dengan halaman pelatihan dengan memiliki tombol masukan untuk *file* dokumen dan tombol untuk proses *lemmatization* dimulai. Halaman ini juga memiliki *textarea* yang digunakan untuk menampilkan isi asli dan hasil dari *lemmatization* untuk dokumen tersebut. Halaman ini juga memiliki 2 tombol HAM dan SPAM untuk memberikan inisialisasi awal pada dokumen tersebut. Halaman ini juga memiliki tombol uji data untuk melanjutkan proses pelatihan.

5.2 Hasil Lemmatization

Lemmatization adalah cara untuk mendapatkan kata dasar dengan menghilangkan berbagai imbuhan dan mengacu pada kamus yang disediakan untuk hasil yang lebih akurat.

Adapun 4 jenis imbuhan yang utama dalam bahasa Indonesia yaitu akhiran partikel, akhiran *possessive pronouns* atau kepunyaan, *derivational suffix* (akhiran), dan *derivational prefix* (awalan). Adapun imbuhan yang berupa akhiran partikel adalah (-kah, -lah, -tah, dan -pun). Adapun imbuhan yang berupa akhiran *possessive pronouns* adalah (-ku, -mu, dan -nya). Adapun imbuhan yang berupa *derivational suffix* adalah (-i, -kan, dan -an). Adapun imbuhan yang berupa derivational prefix adalah untuk tipe *complex* (be-, me-, pe-, dan te-) dan untuk tipe *plain* (di-, ke-, se-).

Adapun penerapan *lemmatization* terletak setelah memasukkan *file* dan *file* tersebut akan diolah melalui proses *preprocessing* selain *lemmatization.* Proses ini dimulai dengan mengambil kata-kata dari tabel *word* untuk pelatihan dan tabel *word filter* untuk pengujian. Setelah melewati *class* berdasarkan OOP. Maka hasilnya akan disimpan dalam tabel *result* untuk pelatihan dan tabel *resulfilter* untuk pengujian.

Pertama yang dilakukan adalah melihat ketersediaan kata pada kamus untuk kata yang diambil sebagai *input* dari tabel *word* dan *wordfilter*. Terdapat beberapa kondisi yang dilihat saat mengecek apakah kata tersebut ada dalam kamus adalah:

1. Jika jumlah huruf kurang dari 3 maka menghasilkan *false/* tidak diproses.
2. Jika terdapat kata yang berulang maka dipilih salah satu.
3. Jika kata diakhiri dengan *vowel*(aiueo) dan *derivational suffix* adalah -kan ada kemungkinan seperti *overstemming*, maka algoritma akan mengecek untuk kedua kemungkinan dengan -k atau tanpa -k.

Selanjutnya setelah kata tidak tersedia dalam kamus maka akan dilakukan proses mengecek kata inputan untuk aturan presisi; jika inputan kata memliki sebuah "confix" be - lah, be - an, me - i, di - i, pe - i, te - i maka penghapusan *derivational prefix* akan dilakukan pertama.

Selanjutnya jika kata tidak terkait aturan presisi maka akan dihapus sesuai imbuhan yang ada pada kata tersebut. Dalam hal ini terdapat 3 fungsi untuk melakukan pengapusan imbuhan yaitu *delete inflection suffix* (*particle* dan

possessive pronouns, *delete derivational suffix*, dan *delete derivational prefix*. Semua fungsi ini memiliki bentuk yang sama terkecuali untuk *derivational prefix*.

Kecepatan *lemmatization* dapat diukur dengan mencari kecepatan rata-rata dokumen secara acak dan diutamakan yang berisikan banyak kata-kata. Kecepatan rata-rata yang didapatkan setelah menguji 10 dokumen secara acak yang dilakukan *lemmatization* adalah 0.86 detik. Hasil dari pengujian dapat dilihat pada tabel 6.

Tabel 6. Tabel Kecepatan *Lemmatization*

NO	Nama *File*	Ukuran	Kecepatan
1	0021.1999-12-15.kaminski.ham.txt	2 Kb	2.57398 detik
2	4358.2005-07-07.SA_and_HP.spam.txt	1 Kb	0.75479 detik
3	0027.1999-12-16.kaminski.ham.txt	1 Kb	0.38865 detik
4	4362.2005-07-07.SA_and_HP.spam.txt	1 Kb	0.57081 detik
5	0029.1999-12-16.kaminski.ham.txt	2 Kb	1.71782 detik
6	4374.2005-07-07.SA_and_HP.spam.txt	1 Kb	0.91123 detik
7	0031.1999-12-16.kaminski.ham.txt	1 Kb	0.71063 detik
8	4380.2005-07-07.SA_and_HP.spam.txt	1 Kb	0.58566 detik
9	0035.1999-12-16.kaminski.ham.txt	1 Kb	0.67463 detik
10	4385.2005-07-11.SA_and_HP.spam.txt	1 Kb	0.81843 detik

5.3 Pengujian

Proses perhitungan manual yang dilakukan oleh *filtering email SPAM* dengan menggunakan metode *naive bayes classifier* dan *lemmatization* akan dijelaskan pada sub bab ini. Sebagai contoh, *email* yang digunakan adalah *email* yang bernama *4702.2005-07-15.SA_and_HP.SPAM.txt* dengan inisialisasi *value* adalah *SPAM*.

1. Pertama mencari nilai probabilitas dari keseluruhan kategori pada data latih, berikut perhitungan nilai probabilitas dari keseluruhan kategori berdasarkan persamaan (7).

$$P(SPAM) = \frac{jumlah\ banyaknya\ SPAM\ pada\ data\ latih}{Jumlah\ keseluruhan\ data\ latih}$$

$$P(HAM) = \frac{jumlah\ banyaknya\ HAM\ pada\ data\ latih}{Jumlah\ keseluruhan\ data\ latih}$$

$$P(SPAM) = \frac{201}{510} = 0{,}39411765$$

$$P(HAM) = \frac{309}{510} = 0{,}60588235$$

2. Selanjutnya memberikan status kata-kata yang terdapat dalam *file email* tersebut pada tabel *trained*. Terdapat 3 status yaitu data latih, bukan data latih, *stopword*. Status data latih adalah kata-kata yang masuk dalam data latih yang telah dilatih sebanyak 3175 kata yang terdapat dalam tabel *trained* dan memiliki *SPAM* probabilitas dan HAM probabilitas. Status bukan data latih adalah kata yang tidak termasuk dalam pelatihan data pada tabel *trained*. Status *stopword* adalah kata-kata yang terdapat pada Lampiran *stopword*.

Tabel 7. Status Pengujian

No	Kata	Status
1	Obat	Data Latih
2	Generik	Data Latih
3	Resep	Data Latih
4	Baik	*Stopword*
5	Kurang	*Stopword*
6	Mata	*Stopword*
7	Yang	*Stopword*
8	Siap	*Stopword*
9	Adalah	*Stopword*
10	Tanda	Data Latih
11	Khianat	Bukan data latih
12	Bukan	*Stopword*
13	Karena	*Stopword*
14	Sedih	Data Latih
15	Hidup	*Stopword*
16	Adalah	*Stopword*
17	Kebun	Data Latih
18	Binatang	Bukan data latih
19	Hutan	*Stopword*
20	Kita	*Stopword*
21	Dua	*Stopword*
22	Adalah	*Stopword*
23	Untuk	*Stopword*
24	Diri	*Stopword*
25	Kita	*Stopword*

Tabel 8. Kata Dalam Data Latih

No	Kata	*SPAM* Probability	Ham Probability
1	Obat	0.00218793 18286504	0.00047619 047619048
2	Generik	0.00034546 292031322	0.00034546 292031322
3	Resep	0.00069092 584062644	0.00011904 761904762
4	Tanda	0.00034546 292031322	0.00119047 61904762
5	Sedih	0.00023030 861354215	0.00011904 761904762
6	Kebun	0.00034546 292031322	0.00034546 292031322

Kata yang termasuk dalam status data latih merupakan kata yang masuk dalam perhitungan *naive bayes classifier*. Kata yang termasuk dalam status *stopword* dan status yang bukan data latih tidak masuk ke dalam proses perhitungan untuk pengujian.

3. Kemudian mengitung hasil prediksi dengan rumus *naive bayes classifier* berdasarkan tabel diatas. Hasil dari tabel diatas akan dibulatkan. Setelah itu dibandingkan nilai dokumen yang sama untuk setiap kategori. Berikut perhitungan perhitungan probabilitas dokumen yang sama untuk setiap kategori berdasarkan persamaan (5).

$$P(4702.2005-07-15.SA\_and\_HP.spam.txt \mid HAM) = P(HAM) \times P(obat|HAM) \times P(generik|HAM)\, P(resep|HAM) \times P(tanda|HAM) \times P(sedih|HAM) \times P(kebun|HAM)$$

$$P(4702.2005-07-15.SA\_and\_HP.spam.txt \mid HAM) = (0{,}6059) \times (0.0005) \times (0.0003) \times (0.0001) \times (0.0001) \times (0.0001) \times (0.0003) = 2{,}72655\times10^{-23}$$

$$P(4702.2005-07-15.SA\_and\_HP.spam.txt \mid SPAM) = P(SPAM) \times P(obat|SPAM) \times P(generik|SPAM) \times P(resep|SPAM) \times P(tanda|SPAM) \times P(sedih|SPAM) \times P(kebun|SPAM)$$

$$P(4702.2005-07-15.SA\_and\_HP.spam.txt \mid SPAM) = (0{,}3941) \times (0.0022) \times (0.0003) \times (0.0007) \times (0.0003) \times (0.0002) \times (0.0003) = 3{,}2773356\times10^{-21}$$

$$P(3{,}2773356\times10^{-21}) > P(2{,}72655\times10^{-23})$$

$$P(4702.2005-07-15.SA\_and\_HP.spam.txt \mid SPAM) > P(4702.2005-07-15.SA\_and\_HP.spam.txt \mid HAM)$$

Jadi hasil yang didapat setelah melakukan adalah *4702.2005-07-15.SA_and_HP.SPAM.txt* berhasil terklasifikasi menjadi *SPAM*. Ini dikarenakan hasil perhitungan probabilitas untuk kategori *SPAM* lebih besar dibandingkan HAM.

5.4 Hasil Penelitian

Jumlah *file* untuk data pelatihan adalah sebanyak 510 *file* yang terbagi atas 309 *file* HAM dan 201 *file SPAM*. Total jumlah kata yang diproses adalah berjumlah 3175 kata dengan pembagian 5507 kata yang berada pada *file SPAM* dan 5223 yang berada pada *file* HAM.

Berdasarkan pelatihan yang telah selesai dilakukan, terdapat 5 kata yang paling sering ditemukan pada *dataset* ini untuk masing-masing kategori. 5 kata teratas berturut-turut untuk kategori *SPAM* adalah vince, *enron*, temu, risiko, dan model. Sedangkan 5 kata teratas berturut-turut untuk kategori HAM adalah *adobe*, *email*, kirim, tarik, dan situs.

Setelah melakukan pelatihan, data diuji dan dianalasis hasilnya. Data uji berjumlah 131 yang terdiri atas 63 *file* awal HAM dan 68 *file* awal *SPAM* diuji dengan membandingkan dengan data latih.

Hasil pengujian disajikan dalam bentuk *confusion matrix*. Berikut adalah *confusion matrix* hasil pengujian:

Tabel 9. *Confusion Matrix* Hasil Pengujian Data *(Angka)*

Nilai Asli	**Prediksi**	
	HAM	**SPAM**
HAM	63	0
SPAM	12	56

Tabel 10. *Confusion Matrix* Hasil Pengujian Data *(*persentase*)*

Nilai Asli	**Prediksi**	
	HAM	**SPAM**
HAM	100%	0%
SPAM	19,05%	82,35%

Berdasarkan kedua *confusion matrix* diatas, didapatkan hasil sebagai berikut:

1. $Accuracy = \frac{TP+TN}{N(testing)} = \frac{63+56}{131} = \mathbf{90,83\%}$
2. $Error\ Rate = 1 - Accuracy = \mathbf{9,17\%}$
3. $Precision = \frac{TP}{TP+FP} = \frac{63}{63+12} = \mathbf{84\%}$
4. $Recall = \frac{TP}{TP+FN} = \frac{63}{63+0} = \mathbf{100\%}$

Hasil dari proses klasifikasi *email* yang didapat memiliki akurasinya yang tinggi, yaitu 90,83% dengan nilai *true positives* (*true* HAM) adalah 100% dan nilai *true negatives* (*true SPAM*) adalah 82,35%.

6. KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, maka dapat diambil simpulan sebagai berikut:

1. Kecepatan rata-rata *lemmatization* pada saat memisahkan kata dasar adalah 0.86 detik berdasarkan Tabel 5.1.
2. Kecepatan rata-rata *lemmatization* pada saat memisahkan kata dasar adalah 0.86 detik.Pada *file 0021.1999-12-15.kami nski.ham.txt* yang memiliki ukuran file 2 Kb melakukan *lemmatization* selama 2.57398 detik. Kecepatan *lemmatization* berbanding lurus dengan ukuran *file email* .Ini dapat dilihat pada Tabel 5.1, *file* yang memiliki ukuran yang lebih besar membutuhkan waktu yang lama dalam pemrosesan.
3. Klasifikasi dengan menggunakan *algoritma Naive Bayesian* yang diteliti dari total 131 *file*, berhasil mengklasifikasikan dengan benar sebanyak 119 *file* dan 12 *file* yang diuji mendapatkan nilai prediksi yang salah. Sehingga persentase keberhasilan yang didapatkan pada penelitian ini adalah 90,83%.

6.2 Saran

Berdasarkan penelitian yang telah dilakukan oleh penulis, terdapat beberapa hal yang patut diperhatikan dan penulis sarankan agar aplikasi ini dapat dikembangkan lebih pada penelitian selanjutnya:

1. Saat aplikasi sedang melakukan proses penyimpanan pada *database*, aplikasi tidak dapat dijalankan dan cenderung lama dalam pemrosesan. Maka dibutuhkan *source code* yang sederhana untuk mempercepat komputasi.
2. Aplikasi ini tidak menggunakan *term-weighting* dan *feature selection*. Karena tidak ada tahapan pembobotan dan pemilihan fitur maka proses pelatihan cukup memakan waktu, karena tidak ada proses penyeleksian pada kata yang akan dilatih.
3. Aplikasi belum dapat memilih dan menyimpan banyak *file* untuk pelatihan sekaligus, masih satu per satu, sehingga diperlukan pengembangan yang dapat memproses banyak *file* dalam satu waktu, sehingga waktu yang diperlukan dapat dipangkas.
4. Aplikasi ini tidak dapat digunakan jika belum terisi data latih. Maka disarankan untuk melakukan pelatihan data yang banyak supaya persentase keberhasilan semakin tinggi.

DAFTAR PUSTAKA

[1] Taufik, A., 2017. Analisis Filtering Email Spam Dengan Mengguanakn Metode Naive Bayesian. *JURNAL INFORMATIKA DEPARTEMEN ILMU KOMPUTER FAKULTAS MATEMATIKA DAN ILMU*, Volume IIII, pp. 1 -10..

[2] LaMacchia, B. A. & Cranor, L. F., 1998. Spam *Communications of the ACM*, pp. 74-83.

[3] I. R. Widiasari and T. I. Bayu, "Pembangunan Spam E-Mail Filtering System dengan Metode Naive Bayesian," *Konf. Nas. Sist. Inf. 2013*, vol. 72, pp. 284–290, 2013.

[4] Liu, H. & Christiansen, T., 2012. BioLemmatizer: a lemmatization tool for morphological processing of biomedical text. *Journal of Biomedical Semantics*, p. 3.

[5] J. Palme, *Electronic mail*. Boston: Artech House, 1995.

[6] S. Ghiam and A. N. Pour, "A Survey on Web SPAM Detection Methods: Taxonomy," *Int. J. Netw. Secur. Its Appl.*, vol. 4, no. 5, pp. 119–134, 2012.

[7] S. Vijayarani, J. Ilamathi, and M. Nithya, "Preprocessing Techniques for Text Mining - An Overview," *Int. J. Comput. Sci. Commun. Networks*, vol. 5, no. 1, pp.

7–16, 2015.

[8] A. Indriani, "Klasifikasi Data Forum dengan menggunakan Metode Naïve Bayes Classifier," *Semin. Nas. Apl. Teknol. Inf.*, pp. 5–10, 2014.

[9] I. R. Widiasari and T. I. Bayu, "Pembangunan Spam E-Mail Filtering System dengan Metode Naive Bayesian," *Konf. Nas. Sist. Inf. 2013*, vol. 72, pp. 284–290, 2013.