Penerapan Random Forest dalam Email Filtering untuk Mendeteksi spam

Billy Christanto, Djoni Haryadi Setiabudi
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121-131. Surabaya 60236
Telp (031) 2983455, Fax. (031) 8417658
Email: billychrsitanto5@gmail.com, djonihs@petra.ac.id

ABSTRAK

Email sudah menjadi komponen integral bagi pengalaman menjelajah internet. Seiring meningkatnya pengguna, pemasaran lewat email juga semakin populer. Email sejenis ini sering kali dianggap mengganggu oleh pengguna. Karena jumlahnya yang eksesif, maka muncul kebutuhan untuk memisahkan email yang tidak penting dari yang penting. Hingga sekarang, belum ada solusi optimal bagi masalah ini. Dari berbagai metode, solusi berbasis machine learning menunjukan hasil yang paling menjanjikan.

Metode yang diujikan adalah *Random Forest* yang seringkali dianggap superior dibanding *Naïve Bayesian*, salah satu metode populer untuk email filtering. Kedua algoritma akan dibandingkan lewat akurasi, recall dan presisi-nya. Efek *pre-processing* dan *stemming* terhadap dataset juga akan diujikan.

Hasil penelitian menunjukkan bahwa kedua model dapat menghasilkan akurasi, recall dan presisi yang mencapai sekitar 96%. Pengujian juga menunjukkan bahwa model *Random Forest* membutuhkan waktu training sekitar 80 kali lebih lama dari *Naïve Bayes* sehingga kurang sesuai untuk digunakan dalam *email filtering.*

Kata Kunci: *Spam, Email Filtering, Naive Bayes, Random Forest*

ABSTRACT

Email became an integral part of the internet experience. As users increase, marketing via email also became more popular. These emails often annoy users, hence the name "spam". Because of its excessive number, the need to separate important messages from unimportant ones emerges. Up until this point, there's no optimal solution to this problem. Among the methods being used, machine learning based solutions show the most promising results.

The method being tested is Random Forest, which is often regarded as superior compared to Naïve Bayesian, a popular algorithm for email filtering. Both of the algorithms are to be subjected to tests and compared for their accuracy, recall and precision. The effects of pre-processing and stemming to the dataset will also be tested.

This research shows that both models produce similar accuracy, recall and precision that reach 96% for each category. Tests also show that Random Forest needs around 80 times more time to train it's model compared to Naive Bayesian so it became not suitable for email filtering purposes.

Keywords: *Spam, Email Filtering, Naive Bayes, Random Forest.*

1. PENDAHULUAN

Surat elektronik atau yang lebih dikenal dengan nama email, merupakan suatu bagian yang esensial bagi pengguna komersial maupun bisnis. Email juga merupakan komponen integral bagi pengalaman menjelajah internet karena akun email (email address) dibutuhkan untuk mendaftarkan diri dalam aktivitas online, termasuk social network, instant messaging, dan akun lainnya dalam internet. Pada tahun 2019, diperkirakan 293 Miliar email dikirimkan tiap harinya. Pengguna email rata-rata menerima 126 email tiap harinya [9]. Tak dapat disangkal bahwa email sudah menjadi media komunikasi yang sangat dominan bagi pengguna komersial maupun bisnis.

Karena banyaknya pengguna email, seringkali email digunakan sebagai media pemasaran. Email sejenis ini sering dianggap mengganggu dan tidak penting karena tidak diinginkan oleh sebagian besar dari pengguna. Diperkirakan sekitar 55% dari seluruh email yang terkirim dapat dikategorikan sebagai bulk/junk email [9]. Email sejenis sekarang lebih dikenal dengan nama Spam.

Seiring bertambahnya jumlah email muncul juga kebutuhan untuk menyaring dan memisahkan email penting dari yang tidak penting. Solusi tradisional dari masalah ini adalah menggunakan list yang memblokir pengirim tertentu seperti blacklist dan whitelist [8]. Solusi ini sering dianggap tidak efektif karena akurasinya yang semakin lama semakin menurun dan seringkali memblokir email penting yang ditunggu oleh pengguna. Solusi lain adalah menggunakan machine learning. Dengan mempelajari pola dari karakteristik email, spam akan mudah dipisahkan dari email lainnya yang dianggap penting. Dengan menggabungkan machine learning dan metode tradisional, jumlah spam yang masuk dalam inbox dapat ditekan[4].

Algoritma machine learning yang sering dipakai adalah Naive Bayesian. Algoritma sederhana ini dinilai cukup efektif dalam penggunaan email filtering. Tapi, algoritma ini memiliki kelemahan berupa kerentanannya pada overfitting. Di lain sisi, Algoritma Random Forest sering dianggap menjadi algoritma yang lebih superior dibanding Naive Bayesian. Kelemahannya hanya di waktu training yang dibutuhkan oleh Random Forest lebih lama dari Naive Bayesian.

Pada penelitian ini akan diujikan algoritma Random Forest untuk content-based email filtering dan hasilnya akan dibandingkan dengan Naive Bayesian untuk mencari metode yang lebih efektif untuk penerapan dalam email filtering.

2. TINJAUAN STUDI

Topik penggunaan machine learning dalam email filtering sudah tidak asing dalam dunia penelitian. Anugroho [2] sudah menguji algoritma Naive Bayesian dan menemukan bahwa selisih jumlah keyword dalam email spam dan ham dapat mempengaruhi nilai error pada algoritma. Renuka dan Hamsapriya [7] mengujikan penggunaan word-stemming untuk mengekstrak kata-kata penting lame sebuah email agar metode content-based filter dapat bekerja secara lebih efektif. Dengan pengaplikasian stemming, akurasi dari Naive Bayesian dapat ditingkatkan sebesar 5-10%. Lovin [6] mengujikan

berbagai macam algoritma machine learning populer untuk email filtering seperti SVM, kNN, dan Naive Bayesian. Lovin menemukan dari ketiga metode tersebut, SVM menghasilkan akurasi yang paling tinggi tapi dengan waktu training yang paling tinggi juga. Akinyelu dan Adewumi [1] membuat filter untuk mendeteksi phising dengan Random Forest. Presisi yang dihasilkan tinggi dan tingkat kesalahan yang sangat rendah. Bahgat, et al [3] mengaplikasikan WordNet Ontology dan metode semantic lainnya untuk mengurangi jumlah kata yang diproses oleh algoritma sehingga mengurangi waktu prose secara signifikan.

3. DESAIN SISTEM

3.1. Dataset dan Preprocessing

Dataset email berbahasa inggris diambil dari dataset yang telah disediakan pada [5]. Dataset dalam bentuk plain-text. Dataset kemudian dibagi untuk keperluan training algoritma dan testing.

Sebelum digunakan dalam program, data perlu diolah terlebih dahulu untuk diintegrasikan ke dalam program. Berikut adalah beberapa hal yang dilakukan dalam pengolahan data:

1. Mengambil body dari email. Text dalam header tidak akan masuk dalam pertimbangan algoritma sehingga perlu dibuang.
2. Jika body text masih berbentuk html, tag html akan dibuang.
3. Text yang didapat akan diubah menjadi bentuk huruf kecilnya untuk konsistensi.
4. Huruf baca yang ada akan dibuang karena hanya menambah token yang tidak diperlukan dalam email.
5. Kata-kata yang terlalu pendek dan tidak memiliki arti akan dibuang dengan tujuan mengurangi jumlah kata yang diproses oleh program.
6. Dilakukan stemming pada tiap kata untuk mendapat kata dasar dari tiap kata.

Pre-processing tersebut akan memudahkan saat merubah setiap email menjadi matriks kemunculan kata.

3.2. Classifier Naive Bayes

Untuk mengatasi masalah zero frequency, model bayesian yang akan digunakan adalah Multinomial Naive Bayesian. Dengan menggunakan Laplace smoothing, model tetap bisa memprediksi meskipun ada kata-kata baru yang muncul pada tahap testing.
Classifier Naive Bayesian dibentuk dengan cara melakukan training dengan data yang sudah disiapkan. Algoritma akan menghitung probabilitas kata muncul pada kategori tertentu untuk memprediksi kategori email lainnya.

3.3. Classifier Random Forest

Classifier Random forest dibentuk dengan cara membangun decision tree sejumlah yang ditentukan. Algoritma akan membuat bootstrap dataset yang dibentuk secara random. Dataset tersebut lalu digunakan untuk membuat decision tree. Random Forest akan memproses data dengan cara menjalankan tiap data terhadap decision tree yang dibentuk, dari hasil decision tree tersebut akan diambil prediksi mayoritas. Prediksi tersebut akan menjadi hasil akhir dari Random Forest.

3.4. Pengujian

Pengujian akan dilakukan dengan cara memberikan data testing yang sama pada kedua algoritma. Hasil output dari algoritma akan dibandingkan dengan klasifikasi aslinya. Algoritma akan diukur dengan metrik akurasi, recall, dan presisi dari output algoritma.

4. PENGUJIAN SISTEM

4.1. Konfigurasi Pengujian

Pengujian dilakukan terhadap data yang sudah disiapkan. Data tersebut sudah dibersihkan dengan cara yang sama pada proses training. Algoritma akan menghasilkan output berupa nilai klasifikasi email. Email spam akan mendapat nilai 1(satu) dan ham akan mendapat nilai 0(nol). Hasil dari output program akan dibandingkan dengan klasifikasi asli dari email.
Kata-kata tidak penting yang dibuang diambil dari library stopwords dari NLTK. Stemmer yang digunakan adalah Porter Stemmer. Untuk classifier Random Forest, akan dibangun 25 decision tree dengan kriteria split berupa *gini score*.

4.2. Pengujian Awal

Pada pengujian ini, pre-processing dan stemming akan dilakukan pada data sebelum diproses oleh algoritma.

Tabel 1. Hasil Run-1 testing waktu train, akurasi, recall dan presisi dengan preprocessing dan stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.72	96.13	97.25	97.60
Naive Bayes	0.006	97.93	98.40	98.84

Pada tabel 1 dapat dilihat bahwa akurasi, recall dan presisi dari kedua algoritma mirip, dengan Naïve bayes menunjukkan hasil yang sedikit lebih tinggi. Waktu training Random Forest pada run ini juga sangat tinggi dibanding dengan Naïve bayes.

Tabel 2. Hasil Run-2 testing waktu train, akurasi, recall dan presisi dengan preprocessing dan stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.66	97.47	98.93	97.72
Naive Bayes	0.006	97.53	98.23	98.49

Tabel 3. Hasil Run-3 testing waktu train, akurasi, recall dan presisi dengan preprocessing dan stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.71	95.93	96.43	98.09
Naive Bayes	0.006	97.47	97.41	99.18

Hal yang sama terjadi pada run-2 dan run-3 pada pengujian ini seperti yang ditunjukkan oleh tabel 2 dan tabel 3, dengan run-2 menunjukkan hasil yang paling baik untuk Random Forest, Walaupun demikian, waktu training yang digunakan oleh

Random forest masih secara signifikan lebih tinggi dari Naïve bayes.

Dari hasil pengujian diatas, dapat dilihat bahwa dalam kondisi ideal, Random Forest dapat menghasilkan model prediksi yang sama akuratnya dengan Naive Bayesian. Nilai recall dan presisi yang dihasilkan juga tidak jauh berbeda dari Naive Bayesian. Hanya saja Random Forest mendapatkan hasil tersebut dengan waktu training yang secara signifikan lebih lama dari Naive bayesian.

4.3. Pengujian Tanpa Pre-processing

Pengujian ini akan dilakukan tanpa pre-processing tapi menggunakan proses stemming.

Tabel 4. Hasil Run-1 testing waktu train, akurasi, recall dan presisi tanpa pre-processing.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.69	95.33	96.56	97.08
Naive Bayes	0.007	97.73	98.19	98.72

Tabel 5. Hasil Run-2 testing waktu train, akurasi, recall dan presisi tanpa pre-processing.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.78	96.67	97.90	97.54
Naive Bayes	0.008	97.40	97.71	98.70

Tabel 6. Hasil Run-3 testing waktu train, akurasi, recall dan presisi tanpa pre-processing.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.75	95.46	96.99	96.99
Naive Bayes	0.007	98.20	98.23	99.37

Tabel 4, 5, dan 6 menunjukkan hasil yang sama dengan pengujian awal dengan sedikit perubahan yang dapat dimasukkan dalam margin of error. Hal ini menunjukkan bahwa pre-processing pada data tidak banyak berpengaruh dalam akurasi, recall dan presisi jika data yang digunakan untuk training tersaturasi seperti yang dilakukan diatas.

4.4. Pengujian Tanpa Stemming

Pada pengujian ini akan dilakukan pre-processing pada data tapi tidak dilakukan stemming.

Tabel 7. Hasil Run-1 testing waktu train, akurasi, recall dan presisi tanpa stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.79	96.40	97.01	98.22
Naive Bayes	0.008	97.73	97.45	99.01

Tabel 8. Hasil Run-2 testing waktu train, akurasi, recall dan presisi tanpa stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.77	95.67	96.53	97.65
Naive Bayes	0.006	97.80	97.79	99.01

Tabel 9. Hasil Run-3 testing waktu train, akurasi, recall dan presisi tanpa stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.76	96.20	96.55	98.37
Naive Bayes	0.007	97.40	97.70	98.83

Perubahan yang terjadi dibanding pengujian awal tidak signifikan. Seperti yang ditunjukkan pada tabel 7, 8, dan 9, ada sedikit peningkatan dalam waktu training Random Forest.

4.5. Pengujian Tanpa Preprocessing dan Stemming

Pengujian ini dillakukan tanpa pre-processing dan stemming.

Tabel 10. Hasil Run-1 testing waktu train, akurasi, recall dan presisi tanpa pre-processing dan stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.8	94.93	96.02	97.22
Naive Bayes	0.007	97.55	97.34	99.36

Tabel 11. Hasil Run-2 testing waktu train, akurasi, recall dan presisi tanpa pre-processing dan stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.79	95.07	95.77	97.43
Naive Bayes	0.007	97.40	97.40	99.01

Tabel 12. Hasil Run-3 testing waktu train, akurasi, recall dan presisi tanpa pre-processing dan stemming.

	Waktu train (s)	Akurasi (%)	Recall (%)	Presisi (%)
Random Forest	0.78	95.00	95.77	97.43
Naive Bayes	0.008	97.13	96.85	99.26

Tabel 10, 11, dan 12 menunjukkan ada perubahan sekitar 1-2 persen pada akurasi, recall dan presisi random forest. Perubahan ini tidak signifikan dan masih dalam margin of error.

4.6. Perbandingan dengan SpamAssassin

Pada pengujian ini program akan dibandingkan dengan solusi open source yang sering digunakan sebagai mail filter. Pengujian SpamAssassin akan dilakukan dengan bantuan dari API Spam Checker dari PostmarkApp. API akan memberikan skor penilaian spamassassin. Jika skor yang diberikan kurang dari 5.0, email akan dikategorikan sebagai ham. Sebaliknya, email yang mendapat skor lebih dari/sama dengan 5.0, akan dikategorikan sebagai spam.

Pengujian ini akan menggunakan 100 data terpisah dari dataset. 70 diantaranya adalah spam dan 30 sisanya adalah ham. Spamassassin akan menggunakan setting default. Data diambil dari [9] dan [10]. Data-data tersebut dipilih untuk menyesuaikan rasio spam dan ham yang serperti yang dijelaskan sebelumnya.

Tabel 13. Akurasi SpamAssassin, Random Forest, dan Naive Bayesian

	SpamAssasin	Random Forest	Naive Bayesian
Akurasi	70%	86%	96%

Tabel 14. Confusion Matrix Naive Bayesian

Naive Bayesian		Prediction	
		Positive	Negative
Actual	Positive	68	2
	Negative	2	28

Tabel 15. Confusion Matrix Random Forest

Random Forest		Prediction	
		Positive	Negative
Actual	Positive	56	14
	Negative	0	30

Tabel 16. Confusion Matrix SpamAssassin

SpamAssassin		Prediction	
		Positive	Negative
Actual	Positive	42	28
	Negative	2	28

Terlihat dari tabel 13, SpamAssassin secara default tidak se-efektif Random Forest dan Naive Bayesian. Selain itu, SpamAssassin juga menghasilkan nilai false positive yang paling tinggi seperti yang ditunjukkan oleh tabel 16. Perbedaan yang terlihat 2 kali lipat dari Random Forest dan 14 kali lebih dibanding Naive Bayesian seperti yang tertera pada tabel 15 dan 14.

4.7. Pengujian terhadap overfitting

Dalam pengujian ini, model akan diuji akurasinya terhadap data training dan data test. Perbedaan dari kedua akurasi akan menunjukan seberapa tinggi variance yang pada model. Sedangkan akurasi menunjukkan seberapa baik model bisa belajar, semakin tinggi bias, semakin rendah akurasi. Biasanya, bias rendah menunjukkan akurasi yang tinggi yang berarti model bisa belajar dengan baik dari data training. Variance yang tinggi menunjukan kecocokan terhadap data training. Artinya model memiliki generalisasi yang buruk dan kesusahan untuk belajar dari data baru walaupun dengan nilai bias yang rendah. Hal ini yang biasa disebut dengan overfitting.

Tabel 17. Akurasi algoritma terhadap data training dan testing

	Akurasi Data Training	Akurasi Data Testing
Naïve Bayesian	98.49	97.86
Random Forest	99.93	96.40

Dari tabel 17, dapat dilihat bahwa Random Forest memiliki akurasi terhadap data training yang lebih tinggi, menunjukkan bias yang lebih rendah dari Naïve Bayes. Tetapi nilai variance dari Random Forest juga lebih tinggi. Hal ini menandakan bahwa Naïve Bayes dapat menyesuaikan diri pada data baru lebih baik dari Random Forest. Kedua nilai masih dalam batas normal. Dapat disimpulkan bahwa kedua algoritma tidak mengalami overfitting.

4.8. Diskusi

Secara garis besar, dari berbagai pengujian yang dilakukan dapat disimpulkan beberapa hal berikut:

- Dari pengujian awal hingga akhir, hasil yang didapat tidak jauh berbeda antara pengujian satu dengan yang

lain, menandakan bahwa dengan data training sebesar 4500 data, model dapat dianggap stabil. Pre-processing data yang diperlukan hanya membuang header dan tag html jika ada.

- Hasil dari algoritma Naive Bayesian sedikit lebih tinggi dari Random forest dalam semua kasus pengujian. Namun, perbedaan dari kedua algoritma tergolong kecil dan dapat dimasukkan dalam margin of error. Dapat dikatakan bahwa hasil dari kedua algoritma sama.
- Hasil dari content based filtering seperti menggunakan machine learning lebih baik dari score system milik SpamAssassin secara default.
- Tidak ditemukan adanya tanda-tanda overfitting pada kedua classifier. Kedua classifier berjalan sesuai desain dan dapat menyesuaikan dengan data training maupun data testing.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Dari hasil perancangan model dan penelitian ini, dapat diambil beberapa kesimpulan antara lain:

- Kedua algoritma menghasilkan akurasi, recall dan presisi yang mirip jika diberikan data training yang sama. Rata-rata Random Forest menghasilkan akurasi 96.51%, recall 97.54%, dan presisi 97.80. Sedangkan Naive Bayes menghasilkan akurasi 97.64, recall 98.01%, dan presisi 98.83%.
- Proses pre-processing yang diperlukan hanya mengekstrak body text dan membuang tag html. Jika model sudah stabil, pre-processing tidak akan berpengaruh banyak pada hasil akhir. Perbedaan yang dihasilkan hanya 1-2%
- Proses stemming juga tidak berdampak besar pada hasil algoritma, sama seperti pre-processing. Perbedaan yang dihasilkan hanya 1-2%
- Algoritma Random Forest kurang sesuai untuk aplikasi spam filter karena butuh waktu proses yang jauh lebih lama dari naive bayesian. Rata-rata Random Forest membutuhkan waktu 101 kali lebih lama dari Naive Bayes untuk menyelesaikan training dengan data yang sama.
- Algoritma yang diujikan dapat menghasilkan hasil yang lebih baik dari filter tradisional. Random Forest menghasilkan akurasi 86%, Naive Bayes menghasilkan akurasi 96% sedangkan SpamAssassin menghasilkan akurasi 70%.

5.2. Saran

Saran yang diberikan untuk penyempurnaan dan pengembangan lebih lanjut untuk penelitian ini adalah sebagai berikut:

- Melakukan tuning dengan cara mencari jumlah data yang optimal untuk digunakan dalam training model
- Untuk algoritma random forest, dicari jumlah decision tree yang optimal dengan menghasilkan akurasi paling tinggi dengan waktu paling rendah.
- Link dan attachment yang ada dalam email bisa dianalisa untuk mendeteksi email spam

6. DAFTAR PUSTAKA

[1] Akinyelu, Andronicus A., Adewumi, Aderemi O. 2014. Classification of Phishing Email Using Random Forest Machine Learning Technique. Hindawi Publishing Corporation

[2] Anugroho, Prasetyo., Winarno, Idris., Rosyid M., Nur.2010. Klasifikasi email spam dengan metode naive bayes classifier menggunakan java programming.

[3] Bahgat, Eman M., Rdine, Shery, Gad, Wala, Moawad, Ibrahim F. (2018). Efficient email classification approach based on semantic methods. Ain Shams Engineering Journal, 9(4)

[4] Dada, Emmanuel G., Bassi, Joseph S, Chiroma, Haruna, Abdulahaamid, Shafi'i M., Adetunmbi, Adebayo O., Ajibuwa, Opeyemi E. 2019. 'Machine learning for email spam filtering: review, approaches and open research problems'. Heliyon, 5(6).

[5] Enron Spam Dataset. 2006. Retrieved from http://nlp.cs.aueb.gr/software_and_datasets/Enron-Spam/index.html

[6] Loudriyan, Lovin .2014. Analisis kemampuan algoritma-algoritma yang digunakan dalam spam filtering. Universitas Kristen Petra.

[7] Renuka, Karthika., Hamsapriya, T. 2010. Email Classification for spam detection using word stemming. International Journal of Computer Applications 1(5)

[8] Radicati Group.inc 2019. Email Statistic Report. Retrieved from https://www.radicati.com/wp/ wp-content/uploads/2018/12/Email-Statistics-Report-2019-2023-Executive-Summary.pdf

[9] Symantec Corp. 2019. Internet Security Threat Report Vol.24. Retrieved from https://www .symantec.com/content/dam/symantec/docs/reports/istr-24-2019-en.pdf

[10] Guenter, Bruce. 2020. Untroubled Spam Dataset. Retrieved from https://untroubled.org/spam/