Feature Selection pada Phishing Detection dengan Menggunakan Parallel Genetic Algorithm dan Ensemble Learning

Alles Sandro Oktavio Gandadireja, Henry Novianus Palit, Alvin Nathaniel Tjondrowiguno
Program Studi Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-mail: allessandro1999@gmail.com, hnpalit@petra.ac.id, alvin.nathaniel@petra.ac.id

ABSTRAK

Situs *phishing* dapat menjadi ancaman yang mendapatkan informasi pribadi tanpa sepengetahuan pemilik informasi. Setiap situs memiliki catatan secara teknis dalam jumlah yang cukup banyak, yang akan menjadi *feature*. Dikarenakan banyaknya *feature* yang ada, belum tentu semua *feature* tersebut penting bagi pendeteksian situs *phishing*. *Feature selection* menjadi penting pada kasus ini. Penelitian ini menggunakan Genetic Algorithm, dengan Ensemble Learning sebagai fitness function-nya. Dikarenakan untuk menjalankan *feature selection* membutuhkan waktu yang lama, paralelisasi sistem kemudian dilakukan dengan tujuan mempercepat kerja sistem. Hasil dari sistem ini menunjukkan bahwa dengan *feature selection*, terdapat peningkatan akurasi pada klasifikasi. Paralelisasi memberikan bantuan percepatan hingga dua kali lebih cepat. Dengan menggunakan sistem ini, memungkinkan adanya peningkatan keefektifan dalam mendeteksi situs *phishing*.

Kata Kunci: Paralelisasi, Ensemble Learning, Genetic Algorithm, *Feature Selection*, *Phishing*.

ABSTRACT

Phishing sites could become a threat, which retrieves personal information without the user knowing this action. Every site has numerous records, which will be converted to features. Not all features extracted are relevant. Feature selection becomes the main topic of this case. This research uses Genetic Algorithm, using Ensemble Learning as fitness function. This process requires a lot of time, parallelization then used to improve the execution time of the system. The results show that with feature selection, an improvement could be obtained. Parallelization also helps improving execution time up to 2 times faster. Using this system, it is possible to improve the effectiveness of phishing detection.

Keywords: *Parallelization, Ensemble Learning, Genetic Algorithm. Feature Selection, Phishing.*

1. PENDAHULUAN

Pada era modern ini, hampir seluruh data pribadi tersimpan di internet dan untuk menjaga data tersebut, keamanan sangat dibutuhkan untuk melindungi data tersebut. Beberapa ancaman keamanan yang ada secara luas di internet adalah situs yang merupakan *phishing*. *Phishing* merupakan sebuah ancaman yang dapat menyerang pengguna internet dan memiliki dampak yang besar dalam bidang keuangan maupun perlindungan data pribadi. *Phishing* tidak mudah untuk dilacak dan tidak mudah untuk terhindar, karena *phishing* tidak secara langsung digolongkan sebagai *malicious* [10]. *Phishing* dapat secara mudah mencuri data tanpa disadari oleh pengguna, salah satunya melalui situs yang mirip dengan situs asli dengan pengguna memasukkan data pada situs tersebut. Karena situs yang mirip tersebut, seringkali pengguna tidak menyadari bahwa situs tersebut adalah situs *phishing*.

Penyebab susahnya mengatasi *phishing* menurut [6] adalah kemampuan dan pengetahuan manusia yang terbatas dalam memahami konten yang terdapat di *internet*. Seringkali sebuah perusahaan tidak menjadikan keamanan data sebagai yang utama sehingga banyak celah keamanan yang dapat digunakan oleh pelaku *phishing* untuk mengambil data penting tersebut. Kebiasaan dalam mengakses internet juga merupakan salah satu faktor yang berpengaruh. Pengguna tidak mungkin akan selalu memiliki waktu atau kepentingan untuk melihat konten yang diakses, baik itu tipe dari keamanan yang digunakan maupun peringatan dari *browser*. Seiring berkembangnya teknologi, keamanan melawan *phishing* berkembang menjadi semakin canggih, namun *phishing* sendiri juga demikian, sehingga perlu metode yang lebih baik untuk mengidentifikasi situs *phishing* tersebut [10].

Masalah diatas dapat diatasi dengan membentuk sistem untuk pendeteksian situs *phishing*. Untuk pendeteksian situs *phishing* tersebut, salah satu faktor yang dapat mempengaruhi hasil dari pendeteksian adalah *Feature Selection*. *Feature Selection* membantu meningkatkan akurasi dari *phishing detection* dengan mengurangi *feature* yang kurang pengaruh pada prediksi, sehingga *feature* yang penting saja yang digunakan pada *phishing detection* tersebut.

Salah satu penelitian yang menggunakan *feature selection* untuk *phishing detection* dilakukan oleh [1]. Penelitian ini menggunakan Genetic Algorithm sebagai metode *feature selection*. Genetic Algorithm merupakan *evolutionary-based feature selection* yang dapat mencari *feature* yang optimal secara efektif untuk membantu menghasilkan *classification* yang lebih baik dibandingkan dengan menggunakan *filter-based feature selection*. Namun sebagai pertimbangannya adalah mengorbankan kecepatan komputasi. Hasil dari penelitian ini baik, namun metode tersebut membutuhkan waktu yang lama untuk melakukan *feature selection* [1]. Penelitian mengenai *feature selection* dilakukan oleh [5]. Penelitian ini mengimplementasikan Genetic Algorithm secara *parallel* dengan tujuan untuk mempercepat waktu komputasi tanpa mengurangi akurasi pemilihan *feature*. Penelitian ini memberikan hasil bahwa Parallel Genetic Algorithm

dapat mempercepat proses *feature selection* dan optimization pada SVM sebesar 4x pada beberapa *dataset* daripada menggunakan Genetic Algorithm dan SVM biasa [5]. Penelitian mengenai ensemble learning dilakukan oleh [9]. Peneliti ini menggunakan ensemble learning sebagai metode *classification*. Ensemble learning menggabungkan beberapa *classifier* dengan tujuan untuk meningkatkan akurasi dari prediksi dan mengurangi bias dan varians. *Classifier* yang digunakan pada penelitian ini adalah Gaussian Naive Bayes, Support Vector Machine, K-Nearest Neighbor, Logistic Regression, Multilayer Perceptron (MLP), Gradient Boosting dan Random Forest *Classifier*. Hasil yang didapatkan cukup baik dalam mengatasi *overfitting* yang didapatkan pada *classifier* tertentu [9].

Dengan menggunakan metode *parallel* pada proses *feature selection*, masalah waktu komputasi dapat diselesaikan dengan baik, dan dengan menggunakan ensemble learning, akurasi *learning* dapat ditingkatkan. Pada penelitian ini, *feature selection* dengan menggunakan Parallel Genetic Algorithm-Ensemble Learning akan diterapkan untuk menghasilkan *feature* yang lebih optimal untuk kebutuhan *classification* pada tahap selanjutnya.

2. DASAR TEORI

2.1 *Phishing*

Phishing merupakan sebuah penipuan melalui pencurian data oleh pihak yang tidak berwenang dengan tujuan untuk menyalahgunakan data tersebut. Situs *phishing* umumnya meminta untuk melengkapi data pribadi yang penting, misalnya nomor pengguna akun, *username* dan *password*, informasi kartu kredit, dan informasi *internet banking* [2]. Beberapa *link* yang berada pada situs tersebut mengarah ke situs lain merupakan situs palsu dengan tujuan untuk menyimpan data tersebut.

Menurut [4], terdapat dua tipe umum dari *phishing*: deceptive *phishing* dan malware-based *phishing*.

- *Deceptive-phishing* berhubungan dengan *social engineering*, yang bergantung pada klaim palsu yang terlihat seperti pihak berkepentingan yang sesungguhnya. Melalui metode ini, pelaku mengarahkan pengguna pada situs palsu yang meminta untuk memasukkan data yang diinginkan.
- *Malware-based phishing* adalah penggunaan *software* yang dapat mencuri informasi secara langsung melalui celah keamanan pada komputer pengguna. Melalui metode ini, pelaku akan mengarahkan pengguna ke situs palsu atau menuju situs asli dengan penggunaan *proxy*.

2.2 Genetic Algorithm

Genetic Algorithm adalah sebuah *search algorithm* yang berdasarkan pada prinsip seleksi alam dan genetika yang diusulkan oleh J. Holland pada tahun 1970, yang terinspirasi dari evolusi makhluk hidup [7]. Genetic Algorithm dimulai dengan membentuk populasi awal yang terdiri dari individu-individu yang dipilih secara *random*. Dalam satu individu terdapat beberapa gen. Genetic Algorithm akan mencari individu yang paling fit dan akan digabungkan kembali ke populasi awal sehingga dapat menghasilkan populasi baru dengan individu yang lebih baik pada generasi selanjutnya. *Fitness value* dari sebuah individu didapatkan dari mengukur *fitness function* yang telah ditetapkan.

2.3 Parallel Genetic Algorithm

Parallel Genetic Algorithm (PGA) terbentuk oleh dasar *parallel*, yaitu pada umumnya akan membagi tugas menjadi beberapa bagian dan menyelesaikan bagian tersebut secara bersamaan menggunakan *processor* lebih dari satu sehingga waktu komputasi dapat menjadi lebih cepat.

Menurut [3], terdapat tiga tipe utama Parallel Genetic Algorithm: master-slave, fine-grained, dan coarse-grained.

- Master-slave bekerja seperti Genetic Algorithm (GA) umumnya dan tidak mempengaruhi cara kerja algoritma tersebut. Master-slave menggunakan sebuah *global* populasi dan *fitness value* dihitung secara terdistribusi pada banyak processor.
- Fine-grained membagi populasi menjadi subpopulasi dalam jumlah besar yang diatur oleh banyak *processor*. Fine-grained jarang digunakan karena penggunaan *processor* yang terlalu banyak dan *communication cost* yang terlalu besar untuk setiap generation.
- Coarse-grained membagi populasi menjadi beberapa subpopulasi besar dan proses Genetic Algorithm akan dilakukan pada masing-masing subpopulasi tersebut. Setelah beberapa generasi, masing-masing subpopulasi akan melakukan proses migration, yaitu individu dari subpopulasi berbeda akan ditukar dan membentuk subpopulasi baru untuk proses evolusi selanjutnya [5].

2.4 Ensemble Learning

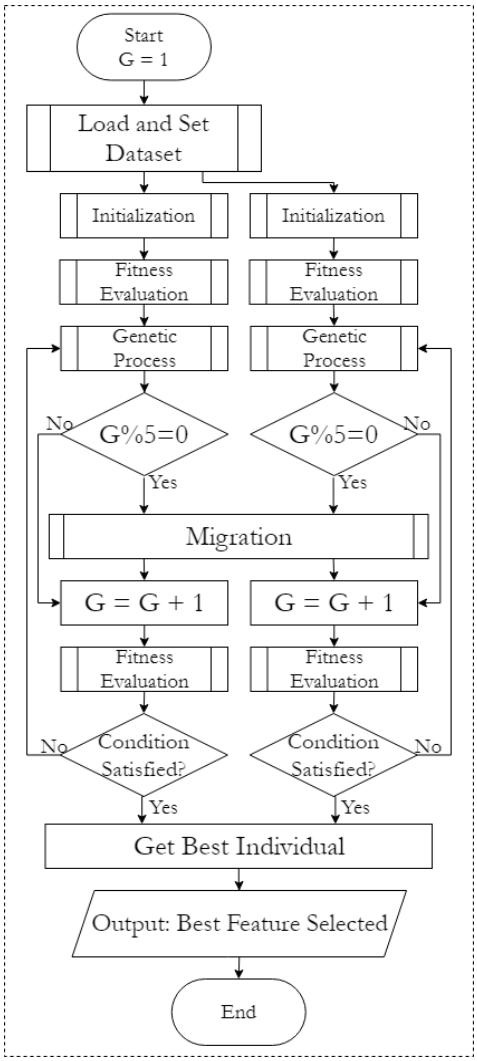
Ensemble learning adalah sebuah metode Machine Learning dimana banyak *learning method* di-*train* untuk menyelesaikan masalah yang sama. Ensemble learning terbentuk dari beberapa *base learner* yang kemudian digabungkan secara sekuensial atau *parallel* dan pada proses *learning* umumnya dipilih dari *majority vote* untuk *classification* dan *weighted average* untuk *regression* [11].

Terdapat tiga metode umum pada ensemble learning: boosting, bagging, dan stacking.

- Boosting dimulai dari memberikan weight pada setiap data training. Kemudian data tersebut dilakukan learning dan dilakukan *classification* pada data test. Untuk setiap data yang salah diprediksi akan diperbesar weight-nya dan kemudian dilakukan learning kembali hingga n kali.
- Bagging dimulai dari membentuk *random* sample dari data *training* dengan *replacement*, lalu data tersebut di-train (membentuk model). Hasil dari setiap model tersebut kemudian digabungkan dengan menggunakan *majority vote*.
- Stacking dimulai dari melakukan training data pada level pertama dengan menggunakan metode *learning* yang berbeda, lalu masing-masing metode *learning* tersebut digabungkan dengan menggunakan *learner* lain pada level kedua, yaitu *meta-learner*.

3. DESAIN SISTEM

Sistem ini menggunakan gabungan dari beberapa metode yang dijelaskan sebelumnya, yaitu Coarse-Grained Parallel Genetic Algorithm, Genetic Algorithm, dan Ensemble Learning. Sistem ini akan berjalan sebanyak k generasi, dimana k akan ditentukan pada pengujian. *Output* dari sistem ini adalah kumpulan *feature* yang memiliki *fitness value* yang paling baik. *Output* yang dihasilkan tidak selalu merupakan *feature* yang terbaik, karena adanya keterbatasan jumlah generasi yang dijalankan dan juga adanya unsur *random* yang digunakan pada Genetic Algorithm ini, yang menghasilkan hasil yang tidak pasti.



Gambar 1. Sistem Genetic Algorithm-Ensemble Learning

Sistem akan berjalan secara parallel, yaitu dengan menggunakan dua proses. Sistem parallel yang dimaksudkan pada penelitian ini adalah Coarse-Grained Parallel Genetic Algorithm (CGPGA). CGPGA memulai dengan membagi jumlah populasi awal menjadi populasi yang lebih kecil yaitu *subpopulation*. Masing-masing *subpopulation* akan menjalankan fungsi *Genetic Algorithm*. Jumlah individu pada *subpopulation* diatur sesuai yang ditetapkan. Penelitian ini akan menggunakan 20 individu yang dibagi menjadi dua menjadi 10 individu pada masing-masing *subpopulation*. *Parameter* yang diterapkan pada masing-masing *subpopulation* akan berbeda satu dengan yang lain, sehingga memungkinkan menghasilkan hasil yang bervariasi.

Gambar 1 menjelaskan tahapan proses dari sistem yang diusulkan. Tahap-tahap yang akan dilakukan pada masing-masing *subpopulation* adalah sebagai berikut:

1. *Subpopulation* akan melakukan proses *initialization*.
2. Perhitungan *fitness value* terhadap masing-masing individu yang telah dihasilkan oleh proses *initialization*, yaitu *fitness evaluation* dengan menggunakan Ensemble Learning sebagai *classifier*. *Classifier* yang akan digunakan adalah k-Nearest Neighbor (kNN), Naïve Bayes (NB), Logistic Regression (LR), dan Random Forest (RF). Hasil dari akurasi Ensemble Learning kemudian akan digunakan untuk menjadi *fitness value* dari sebuah individu.
3. Proses *selection* akan dilakukan terhadap sub*population* yang melakukan *fitness evaluation* terhadap semua individu. *Selection* akan menghasilkan dua individu parent yang terpilih melalui metode yang ditetapkan.
4. Proses *crossover* akan dijalankan pada dua individu parent menghasilkan dua individu *offspring* melalui metode yang ditetapkan.
5. Proses *mutation* akan dijalankan pada dua individu offspring dengan probabilitas terjadi *mutation* dan metode sesuai yang ditetapkan.
6. Proses akan berulang ke proses 2 hingga jumlah generasi tertentu telah tercapai. Jumlah generasi yang akan ditetapkan pada penelitian ini adalah 5.
7. Setelah jumlah generasi yang ditetapkan, proses *migration* akan berjalan.
8. Sistem akan berjalan hingga jumlah generasi maksimum telah tercapai.

Untuk memulai sistem ini, diperlukan proses *initialization*. Pada proses *initialization*, proses yang dilakukan adalah mempersiapkan sejumlah individu yang memiliki *gen* yang berisi 0 atau 1. *Gen* yang memiliki nilai 0 dapat diartikan bahwa *feature* tersebut tidak terpilih dan data akan meng-*exclude feature* yang kemudian dilakukan proses *fitness evaluation*. Sebaliknya, *gen* yang memiliki nilai 1 dapat diartikan bahwa *feature* tersebut terpilih dan akan menjadi salah satu *feature* yang akan dilakukan proses *fitness evaluation*. Nilai pada gen yang dihasilkan akan secara *random*.

Langkah dari proses migrasi yang dilakukan pada sistem ini adalah sebagai berikut:

1. Sebuah process, yaitu P_1, akan menunggu proses lainnya, yaitu P_2, dan sebaliknya.
2. Ketika di antara proses tersebut telah selesai melakukan pekerjaan pada tahap sebelumnya, *fitness value* yang terbaik dari masing-masing proses akan dimasukkan pada sebuah lokasi, yaitu *shared memory*.
3. Pada penelitian ini, shared memory akan berisi sebuah *array* A berukuran 2, dimana masing-masing memiliki *array* berukuran 2, masing-masing A_1 dan A_2. Array A_1 dan A_2 berisi *feature* yang terpilih, berisi 0 dan 1, dan *fitness value* dari *feature* tersebut. Array tersebut akan diisi dengan *fitness value* terbaik pada masing-masing proses.
4. Setelah array A telah diisi, kemudian dilakukan pertukaran diantara dua proses. Proses P_1 akan mendapatkan *feature* terbaik yang dimiliki oleh proses P_2, proses P_2 mendapatkan *feature* terbaik dari proses P_1. *Feature* terbaik tersebut akan menggantikan *feature* yang *fitness value*-nya terburuk pada masing-masing proses.
5. Setelah proses pertukaran selesai, *shared memory* kemudian dihapus isinya untuk digunakan pada proses *migration* selanjutnya.

4. PENGUJIAN SISTEM

4.1 *Dataset*

Sistem ini menggunakan *dataset* yang didapatkan dari UCI Machine Learning dengan nama *Phishing* Websites Data Set [8]. *Dataset* ini memiliki 30 *feature* dan 11.055 *instance*. *Dataset* ini akan digunakan untuk pengujian sistem secara keseluruhan.

4.2 *Testing Environment*

Pengujian dari sistem yang diusulkan pada penelitian ini akan menggunakan tiga buah komputer yang memiliki *processor* yang berbeda. Masing-masing detil *processor* yang akan digunakan pada pengujian ini sebagai berikut:

Tabel 1. Komputer yang digunakan untuk pengujian sistem

Komputer	Source	Operating System	Processor
1	Hosted Runtime	Linux (Virtual Machine)	Intel Xeon vCPU @2.3 GHz
2	Local Runtime	Linux	Intel i7-3770 @3.4 GHz
3	Local Runtime	Linux (Virtual Machine)	Intel i9-9900K @4.7 GHz

4.3 *Parameter* Sistem

Pada penelitian sistem ini, peneliti menetapkan *parameter* yang akan digunakan. Pada setiap sistem berjalan, generasi yang dijalani selalu 50 generasi. *Parameter* yang akan diujikan terdapat pada tabel 2.

Tabel 2. *Parameter* yang digunakan pada sistem yang diusulkan

Parameter	Parameter Terpilih	Parameter Perbandingan
Ukuran Populasi	20 individu (10 individu pada masing-masing process pada paralelisasi)	
Generasi Maksimum	50 generasi	
Tipe *Selection*	Roulette-wheel	Tournament
Tipe *Crossover*	Two-point	
Jumlah Hasil *Crossover*	2 offspring	
Crossover rate	1.0 (selalu terjadi *crossover*)	
Tipe *Mutation*	Single-point	
Mutation rate	0.1 (10% terjadi *mutation*)	

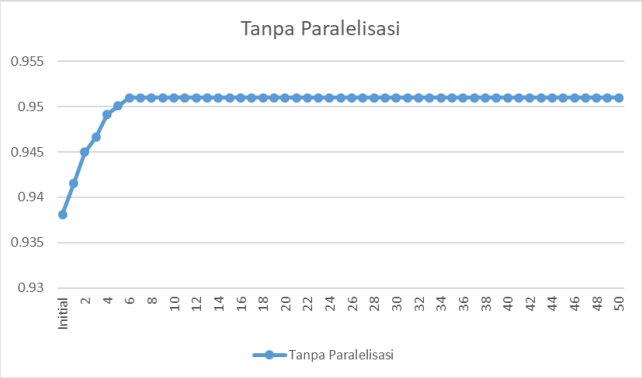
4.4 Hasil Paralelisasi

Dengan atau tanpa adanya paralelisasi, hasil individu terbaik yang didapatkan sistem memiliki *fitness value* yang tidak ada perbedaannya. Tidak ada perbedaan dalam artian fitness value yang didapatkan memiliki nilai yang serupa antara dengan atau tanpa paralelisasi.

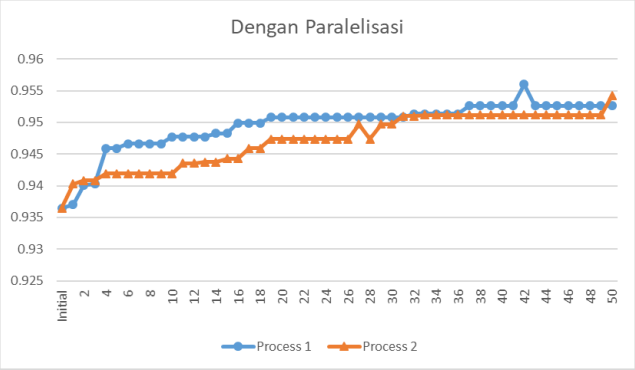
Untuk melakukan pengukuran perbedaan kecepatan antara dengan dan tanpa menggunakan paralelisasi, sistem akan dilakukan modifikasi sesuai dengan sistem yang akan diujikan. Tabel 3 menunjukkan hasil dari pengujian waktu eksekusi. Pengukuran ini adalah hasil dari 50 generasi pada setiap pengulangan. Pengujian dari kecepatan sistem ini dengan menggunakan komputer yang terdapat pada tabel 1.

Dari hasil yang didapatkan, terdapat percepatan berkisar antara 1.8 kali hingga 2 kali lebih cepat pada sistem dengan menggunakan paralelisasi. Hasil ini dapat membuktikan bahwa sistem dengan paralelisasi meskipun melakukan proses *migration* secara *synchronous* tetap memberikan kecepatan yang lebih cepat daripada tanpa menggunakan paralelisasi. Paralelisasi mempercepat proses karena menggunakan *core* dari komputer yang sedang tidak digunakan.

Dengan menggunakan paralelisasi, selain waktu yang didapatkan lebih singkat juga memungkinkan sistem dapat terlepas dari masalah Genetic Algorithm, yaitu *local optima*, walaupun tidak selalu demikian. Dari salah satu hasil yang diambil dari sistem, terdapat perbedaan peningkatan pemilihan individu dengan *fitness value* terbaik antara dengan dan tanpa menggunakan paralelisasi. Perbedaan hasil tersebut terdapat pada gambar 2 dan gambar 3 menggunakan *tournament selection*.



Gambar 2. *Fitness value* tanpa paralelisasi



Gambar 3. Fitness value dengan paralelisasi

4.5 Hasil Pengujian *Parameter*

Metode *selection* yang digunakan pada sistem, yaitu *roulette-wheel* dan *tournament* juga berpengaruh pada waktu yang didapatkan. Hal ini dimungkinkan karena *overhead* yang lebih tinggi pada proses *tournament selection* sehingga menggunakan waktu yang lebih lama. Dari hasil yang didapatkan, terdapat perbedaan sebesar 3% hingga 8% antara roulette-wheel *selection* dan *tournament selection*.

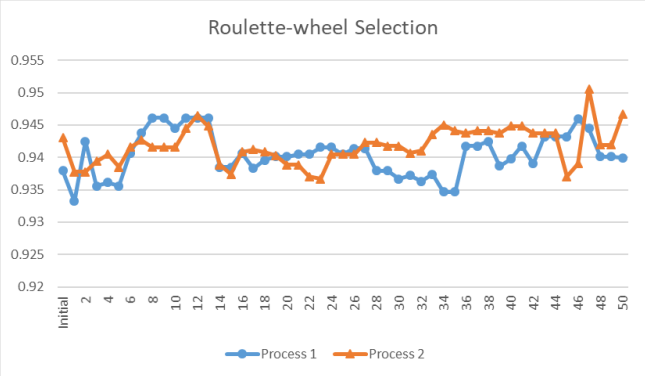
Tabel 3. *Parameter* yang digunakan pada sistem yang diusulkan

#	Tipe *Selection*	Pengujian	# Iterasi	Rata-rata (s)	Percepatan (x)
1	Roulette-wheel	Paralelisasi	38	3741.603	1.17
		Tanpa paralelisasi	5	4390.184	
	Tournament	Paralelisasi	20	3958.985	1.20
		Tanpa paralelisasi	3	4757.898	
2	Roulette-wheel	Paralelisasi	16	2475.923	2.02
		Tanpa paralelisasi	4	4998.771	

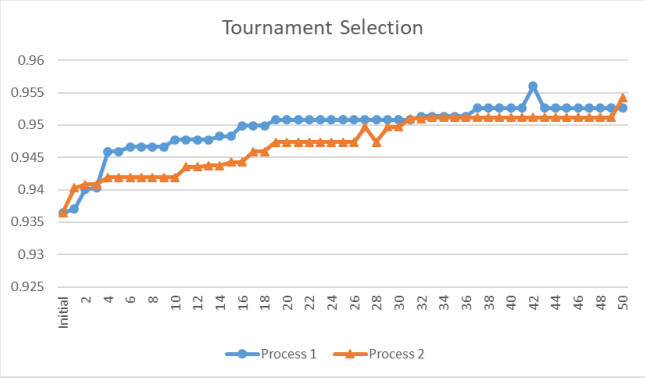
Tabel 3. *Parameter* yang digunakan pada sistem yang diusulkan (sambungan)

#	Tipe *Selection*	Pengujian	# Iterasi	Rata-rata (s)	Percepatan (x)
2	Tournament	Paralelisasi	5	2603.499	1.98
		Tanpa paralelisasi	4	5162.939	
3	Roulette-wheel	Paralelisasi	2	1295.468	1.92
		Tanpa paralelisasi	4	2495.259	
	Tournament	Paralelisasi	2	1407.013	1.82
		Tanpa paralelisasi	2	2563.054	

Selain adanya pengaruh terhadap waktu, akurasi yang didapatkan pada setiap pengulangan juga memiliki perbedaan, yaitu hasil individu yang dihasilkan pada akhir dari sistem. *Tournament selection* cenderung memiliki individu dengan *fitness value* yang lebih tinggi daripada menggunakan *Roulette-wheel selection*. Dari salah satu hasil yang didapatkan dari sistem, dapat terlihat bahwa ada perbedaan antara kedua *parameter* tersebut. Perbedaan hasil tersebut terdapat pada gambar 4 dan gambar 5.



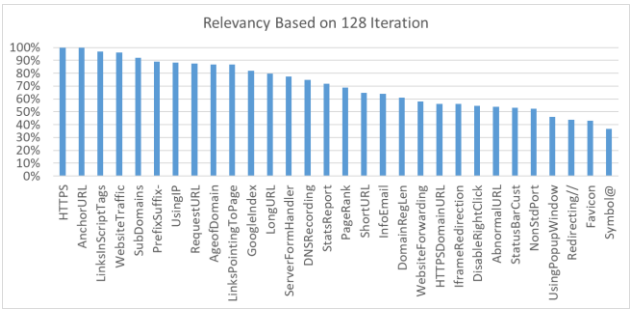
Gambar 4. Sistem menggunakan Roulette-wheel *selection*



Gambar 5. Sistem menggunakan *Tournament selection*

4.6 Hasil Pengujian *Feature Selection*

Sistem yang berjalan 50 generasi tersebut akan diulang selama beberapa kali. Tujuan dari pengulangan tersebut adalah untuk mengukur tingkat relevansi dari masing-masing *parameter*. Semakin tinggi tingkat relevansi sebuah *feature* menunjukkan bahwa hasil perhitungan akan berpengaruh besar ketika *feature* tersebut tidak terpilih. Relenvansi dari *feature* terdapat pada gambar 6.

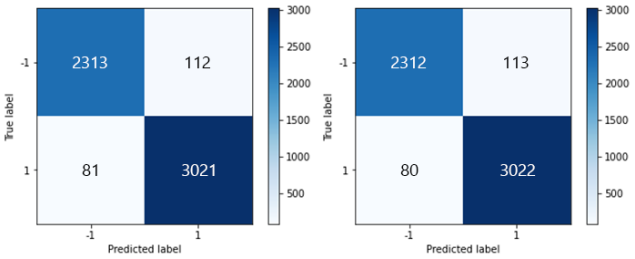


Gambar 6. Sistem Parallel Genetic Algorithm-Ensemble Learning

Hasil yang didapatkan dari sistem memiliki individu terbaik dengan *fitness value* yang berbeda-beda pada setiap eksekusinya. Hal ini dikarenakan adanya *random* pada sistem. Maka dari itu, dilakukan pengambilan kesimpulan dengan melakukan percobaan berulang kali, sehingga mendapatkan *feature* mana saja yang lebih penting, yaitu tingkat relevansi. Dari hasil relevansi tersebut, masih belum dapat diambil kesimpulan untuk *feature* mana saja yang dapat menghasilkan akurasi tertinggi.

Untuk mengambil kesimpulan *feature* yang diperlukan, dilakukan pembatasan, atau threshold. Threshold yang dimaksudkan adalah memilih *feature* yang tingkat relevansinya diatas atau sama dengan threshold yang ditetapkan. Pada penelitian ini, nilai threshold didapatkan dari masing-masing poin dari relevansi. Sebagai contoh dengan threshold 100%, maka *feature* yang terpilih adalah HTTPS dan Anchor URL. Contoh lainnya dengan threshold 37%, maka semua *feature* akan terpilih. Melalui *thresholding*, hasil yang didapatkan adalah sebesar 56%.

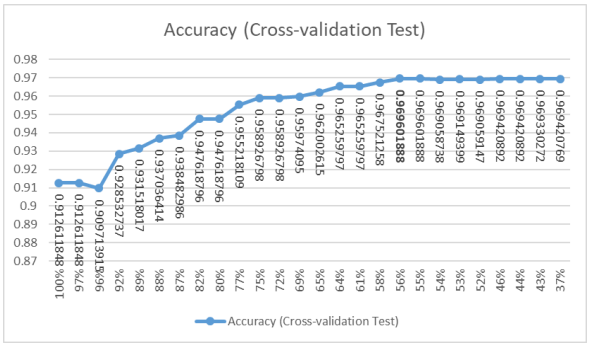
Untuk melihat hasil dari *feature selection*, dilakukan evaluasi berupa *confusion matrix*, yang menunjukkan seberapa akurat proses klasifikasi dilakukan. Pada gambar 7, terdapat perbedaan yang minimal antara dengan dan tanpa paralelisasi.



Gambar 7. Confusion matrix tanpa paralelisasi (kiri) dan dengan paralelisasi (kanan)

Untuk menguji dengan keseluruhan *dataset*, dilakukan pengukuran melalui cross-validation test. Cross-validation test akan dijalankan dua kali, yaitu dengan menggunakan *feature* terpilih dari *feature selection* dan keseluruhan *feature*, dengan k sebesar 5. Dari hasil yang terdapat pada gambar 8, threshold 56% memiliki akurasi yang tertinggi. Perbandingan akurasi klasifikasi antara tanpa *feature selection* dan dengan *feature selection* adalah sebagai berikut:

- Tanpa *Feature Selection*: 0.969149153
- Dengan *Feature Selection*: 0.969601888



Gambar 8. Pengujian *cross-validation* pada seluruh *threshold*

Perbedaan akurasi yang didapatkan adalah sangat kecil. Namun dengan *feature selection*, *feature* yang tidak berguna tidak perlu digunakan.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penggunaan *parameter* yang berbeda memungkinkan perbedaan hasil *feature* yang didapatkan. Namun perbedaan yang ditemui tidak signifikan. *Parameter* yang memungkinkan memiliki pengaruh yang besar dalam pemilihan *parameter* dalam penelitian ini adalah *selection.* Percepatan yang didapat antara tanpa paralelisasi dan dengan paralelisasi pada *feature selection* dengan menggunakan 2 *process* adalah sebesar 1.8 kali hingga 2 kali. Meski terdapat percepatan tersebut, proses *feature selection* ini masih tergolong lama, yaitu sekitar satu hingga dua jam. Dengan *feature selection*, untuk melakukan *feature extraction* dan melakukan klasifikasi akan menjadi lebih cepat daripada tanpa menggunakan *feature selection*.

5.2 Saran

Berdasarkan dari pengujian sistem, dapat diberikan saran yaitu:

- Melakukan pengujian *feature selection* dengan menggunakan *classifier* yang memiliki performa yang lebih cepat dengan akurasi yang baik seperti Random Forest dengan bantuan metode Ensemble Learning lain seperti Bagging dan Boosting.
- Menggunakan metode paralelisasi Genetic Algorithm yang lain, seperti Fine-grained Parallel Genetic Algorithm atau Master-slave Parallel Genetic Algorithm, yang memungkinkan untuk menggunakan seluruh *resource* yang ada untuk mempercepat waktu perhitungan sehingga *feature selection* dapat diselesaikan lebih cepat daripada sistem yang diusulkan saat ini.

6. DAFTAR PUSTAKA

[1] Ali, W. and Ahmed, A.A. 2019. Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting. *IET Information Security*. 13, 6, 659–669. DOI:https://doi.org/10.1049/iet-ifs.2019.0006.

[2] Baykara, M. and Gürel, Z.Z. 2018. Detection of phishing attacks. *6th International Symposium on Digital Forensic and Security, ISDFS 2018 - Proceeding*. 2018-Janua, 1–5. DOI:https://doi.org/10.1109/ISDFS.2018.8355389.

[3] Cantú-Paz, E. 1999. A survey of parallel distributed genetic algorithms. *Complexity*. 4, 4, 31–52. DOI:https://doi.org/10.1002/(SICI)1099-0526(199903/04)4:43.3.CO;2-W.

[4] Chawla, M. and Singh Chouhan, S. 2014. A Survey of Phishing Attack Techniques. *International Journal of Computer Applications*. 93, 3, 32–35. DOI:https://doi.org/10.5120/16197-5460.

[5] Chen, Z. et al. 2016. A parallel genetic algorithm based feature selection and parameter optimization for support vector machine. *Scientific Programming*. 2016. DOI:https://doi.org/10.1155/2016/2739621.

[6] Dhamija, R. and Tygar, J.D. 2005. The battle against phishing: Dynamic security skins. *ACM International Conference Proceeding Series*. 93, 77–88. DOI:https://doi.org/10.1145/1073001.1073009.

[7] Haldurai, L. et al. 2016. A Study on Genetic Algorithm and its Applications. *International Journal of Computer and Engineering*. 4, 10, 139–143. DOI:https://doi.org/10.26438/ijcse/v7i4.184190.

[8] Phishing Websites Data Set: 2015. *https://archive.ics.uci.edu/ml/datasets/Phishing+Websites*. Accessed: 2020-06-12.

[9] Ubing, A.A. et al. 2019. Phishing website detection: An improved accuracy through feature selection and ensemble learning. *International Journal of Advanced Computer Science and Applications*. 10, 1, 252–257. DOI:https://doi.org/10.14569/IJACSA.2019.0100133.

[10] Vayansky, I. and Kumar, S. 2018. Phishing – challenges and solutions. *Computer Fraud and Security*. 2018, 1, 15–20. DOI:https://doi.org/10.1016/S1361-3723(18)30007-1.

[11] Zhou, Z.-H. 2012. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC Press.