Rancang Bangun Model *Machine Learning* untuk Mendeteksi *Malicious Webpage* dengan Metode Wang, *et al.* (2017)

Mohamad Arifandy[1)], Septia Ulfa Sunaringtyas[2)]
(1) Keamanan Siber, Politeknik Siber dan Sandi Negara, mohamad.arifandy@poltekssn.ac.id
(2) Keamanan Siber, Politeknik Siber dan Sandi Negara, septia.ulfa@poltekssn.ac.id

Abstrak

Penggunaan internet dengan mengakses suatu web kerap kali dilakukan untuk berbagai kepentingan. Akibatnya terdapat pihak tertentu yang memanfaatkannya untuk mendapatkan keuntungan. Keuntungan tersebut dilakukan dengan melakukan tindak kejahatan berupa penyisipan konten berbahaya pada suatu halaman web sehingga halaman tersebut dapat dikatakan sebagai malicious webpage. Machine learning yang saat ini sedang menjadi trend dapat digunakan untuk menanggulangi hal tersebut. Machine learning dapat digunakan sebagai cara untuk mendeteksi malicious webpage dengan melakukan klasifikasi terhadap suatu web berdasarkan ciri berupa fitur yang dimiliki dari halaman web. Performa terbaik dari machine learning dalam mendeteksi malicious webpage sangat diperlukan. Pada penelitian ini akan dilakukan rancang bangun model machine learning menggunakan metode Wang, et al. (2017) untuk mendeteksi malicious webpage. Hasil penelitian menunjukkan rancang bangun model machine learning terdiri dari tahapan mempersiapkan lingkungan, pembuatan dataset, dan training dan testing dataset menggunakan algoritme decision tree. Model machine learning memiliki performa berupa akurasi, precision, dan f-measure yang dihasilkan adalah 0.921, 0.925, dan 0.914. Penggunaan algoritme decision tree memberikan performa terbaik dibandingkan dengan menggunakan algoritme lain seperti naïve bayes dan support vector machine (SVM). Performa berupa akurasi, precision, dan f-measure yang dihasilkan dari algoritme naïve bayes adalah 0.738, 0.645, dan 0.773. Performa berupa akurasi, precision, dan f-measure yang dihasilkan dari algoritme SVM adalah 0.802, 0.738, dan 0.807.

Kata kunci: *decision tree, machine learning, malicious webpage, naïve bayes,* performa, SVM.

1. PENDAHULUAN

Pengguna internet di Indonesia tahun 2018 diketahui memiliki presentase 64,8% dari total populasi penduduk Indonesia, naik 10% dibanding tahun 2017 [1]. Umumnya pengguna internet mengakses website untuk berbagai kepentingan seperti mencari informasi, melakukan perdagangan online, hingga berkomunikasi melalui jejaring sosial [2]. Berdasarkan laporan tim Securi, yang didalamnya termasuk *Incident Response Team* (IRT) dan *Malware Research Team* (MRT) menyebutkan bahwa tahun 2018 terdapat 25.466 *website* terinfeksi [3]. Diketahui bahwa terdapat beberapa distribusi malware yang mengakibatkan *website* terinfeksi. Berdasarkan laporan tim tersebut, malware dengan kemampuan *backdoor* berada di urutan teratas dengan presentasi lebih dari 60%, sementara *deface* dan *phising* berada diurutan 7 dan 8 dengan presentasi 10% dan 8.9% dari 9 distribusi malware yang teridentifikasi.

Dampak yang timbul dari *website* terinfeksi tidak hanya membahayakan pengunjung dari *website* tersebut, melainkan dapat menyebar ke komputer lain dari komputer korban yang terhubung dalam satu jaringan yang sama. Komputer korban dijadikan jalur bagi penyerang untuk menyebarkan infeksinya ke komputer lain dan mendapatkan informasi penting [2]. Suatu web terinfeksi terjadi karena dieksploitasi dengan memanfaatkan kerentanan pada sisi client yang kemudian menempatkan konten berbahaya melalui halaman web, sehingga halaman pada web tersebut disebut *malicious webpage* [4]. *Malicious webpage* dapat tergambarkan melalui *phising sites*, *trojan web* [5], ataupun adanya penyisipan *malicious code* melalui XSS pada halaman web [6].

Salah satu upaya yang dapat dilakukan untuk mendeteksi *malicious webpage* adalah dengan memanfaatkan *machine learning*. *Machine learning* ini digunakan sebagai teknik untuk mengklasifikasikan suatu halaman web berdasarkan hasil ekstraksi fitur pada halaman web. Fitur halaman web tersebut digunakan sebagai data yang akan dilatih dengan algoritme tertentu sehingga menghasilkan suatu model yang dapat memprediksi suatu web dikategorikan *malicious webpage* atau tidak [5]. Model *machine learning* tersebut dapat diimplementasikan pada browser yang bekerja dengan mengekstraksi fitur dari halaman web yang akan dikunjungi kemudian model *machine learning* akan mengevaluasi situs yang akan dikunjungi aman atau tidak [6]. Fitur pada halaman web tersebut terdiri dari fitur URL, HTML, ataupun JavaScript. Fitur yang dipilih tersebut harus dapat menggambarkan karakteristik dari *malicious webpage* [7].

Penelitian mengenai penggunaan *machine learning* untuk mendeteksi *malicious webpage* sudah ada sejak beberapa tahun terakhir. Para peneliti membuat model *machine learning* menggunakan cara masing-masing dengan melakukan perbedaan tertentu, seperti jumlah fitur yang digunakan untuk dijadikan dataset, algoritme *machine learning* yang digunakan, serta pengolahan data yang digunakan. Perbedaan tersebut dilakukan untuk menemukan nilai evaluasi dari model atau performa model *machine*

learning yang dihasilkan. Ma, *et al.* (2009) menghasilkan performa terbaik dari model yang menggunakan algoritme *support vector machine* (SVM) dan dataset hanya dari fitur URL [8]. H.B. Kazemian dan S. Ahmed (2015) menunjukkan *machine learning* jenis *supervised* baik untuk mendeteksi *malicious webpage* dibanding jenis *unsupervised* [6]. Dharmaraj R. Patil dan J.B. Patil (2017) menunjukkan penggunaan fitur dan algoritme tertentu berpengaruh terhadap performa yang dihasilkan, termasuk penggunaan algoritme *decision tree, naïve bayes*, dan SVM [9]. Sandag, *et al.* (2018) menunjukkan model dengan menggunakan dataset yang sudah tersedia di internet [10]. Wang, *et al.* (2017) membuat metode gabungan statis dan dinamis untuk mendeteksi *malicious webpage* dengan menerapkan *machine learning* pada tahap statisnya. Penelitian ini menggunakan 17 fitur yang terdiri dari URL, HTML, dan JavaScript serta dapat digunakan untuk mendeteksi halaman web yang memiliki kode berbahaya dalam keadaan diobfuskasi. Model tersebut menggunakan algoritme *decision tree* dan menghasilkan *precision* dan *f-measure* yang tinggi, yakni 92.8% dan 82.8% [5].

Berdasarkan beberapa penelitian yang telah dilakukan tersebut, diketahui bahwa model *machine learning* Wang, *et al.* (2017) memiliki keunggulan antara lain fitur dan algoritme yang digunakan. Fitur yang digunakan pada penelitian tersebut terdiri dari kategori URL, HTML, dan JavaScript, serta dapat digunakan untuk mendeteksi *malicious webpage* meskipun halaman web tersebut memiliki kode yang diobfuskasi, serta algoritme yang digunakan adalah *decision tree*. Keunggulan tersebut menghasilkan performa yang tinggi berupa nilai *precision* sebesar 92.8% dan nilai *f-measure* sebesar 82.8%. Pada penelitian ini dilakukan rancang bangun model *machine learning* menggunakan metode Wang, *et al.* (2017) dan menambahkan penggunaan algoritme lain untuk melihat performa yang dihasilkan dari penggunaan algoritme berbeda.

2. LANDASAN TEORI

2.1. *Machine Learning*

Machine learning merupakan teknik yang digunakan untuk melakukan pemrosesan terhadap data untuk menyelesaikan permasalahan. *Machine lerning* menggunakan algoritme yang secara iteratif belajar dari data untuk menjelaskan ataupun memprediksi hasil dari pemrosesan yang dilakukan [11]. Penggunaan *machine learning* diaplikasikan dengan memproses data menjadi suatu model dengan kegunaan tertentu, seperti akurasi prediksi tingkat tinggi. Model *machine learning* tersebut dapat bersifat prediktif agar dapat membuat prediksi untuk kejadian yang akan datang, atau bersifat deskriptif untuk mendapatkan informasi dari data [12]. *Machine learning* berjenis *supervised* akan digunakan untuk mendeteksi *malicious webpage* karena diharapkan dapat memprediksi kondisi yang akan datang. Performa yang dihasilkan untuk mendeteksi *malicious webpage* dapat diambil dari evaluasi *machine learning* berdasarkan *confusion matrix*. Algoritme *machine learning* yang digunakan antara lain :

- *Decision Tree*

Decision tree merupakan algoritme pembelajaran yang memanfaatkan fungsi Boolean karena dapat membagi suatu node sehingga nantinya dapat merepresentasikan suatu struktur pohon [13]. *Decision tree* digunakan sebagai metode klasifikasi yang merepresentasikan stuktur pohon berisi alternatif – alternatif untuk pemecahan suatu masalah. Peran dari pohon tersebut sebagai *tool* dalam pengambilan keputusan. Dengan adanya *decision tree*, proses pengambilan keputusan kompleks akan menjadi lebih sederhana. Proses utama yang digunakan oleh *decision tree* sendiri adalah mengubah data menjadi keputusan model pohon, mengubah model menjadi *rule*, dan menyederhanakan *rule* [14]. Decision tree termasuk dalam jenis supervised yang menggunakan struktur heirarki. Proses yang dilakukan dimulai dari root node hingga leaf node dan dilakukan secara rekrusif. Setiap percabangan menyatakan kondisi yang harus dipenuhi dan pada setiap ujung pohon menyatakan kelas dari suatu data.

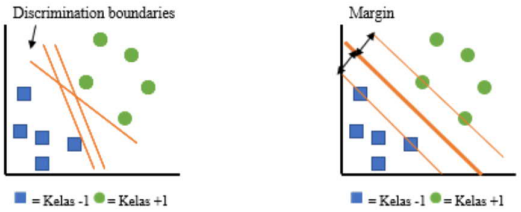
- *Naïve Bayes*

Naïve bayes merupakan algoritme *machine learning* yang termasuk dalam jenis *supervised* dan digunakan untuk klasifikasi. Algoritme ini merupakan teknik prediksi berbasis probabilistik sederhana yang didasari pada penerapan teorema bayes. Ciri utama dari *naïve bayes* adalah asumsi yang sangat kuat (naif) akan independensi dari masing – masing kondisi atau kejadian. Makna dari independensi tersebut adalah bahwa sebuah fitur dalam suatu data tidak akan berkaitan dengan ada atau tidaknya fitur lain dalam data yang sama [14]. *Naïve bayes* sendiri bekerja dengan memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya. Algoritme ini juga diyakini memiliki keutungan dengan hanya membutuhkan jumlah data *training* yang tidak terlalu banyak untuk menentukan estimasi parameter yang diperlukan dalam proses pengklasifikasian. Selain itu, *naïve bayes* juga memiliki keuntungan lain yakni bekerja lebih baik dalam kebanyakan situasi dunia nyata yang kompleks dari pada yang diharapkan [15].

- *Support Vector Machine* (SVM)

Support Vector Machine (SVM) merupakan salah satu algoritme *machine learning* yang menerapkan *pattern recognition*, bertujuan untuk menemukan *hyperplane* terbaik yang memisahkan dua buah kelas pada *input space*. *Pattern recognition* sendiri merupakan istilah dalam melakukan pemetaan suatu data dalam konsep tertentu seperti kelas atau kategori. Konsep dasar SVM merupakan kombinasi dari teori –

teori komputasi yang telah ada sebelumnya. Berbeda dengan strategi *neural network* yang mencari *hyperplane* pemisah antar kelas, SVM berusaha menemukan *hyperplane* terbaik pada *input space* [16].



Gambar 1. Penentuan *Hyperlink* Terbaik [14]

Berdasarkan Gambar 1, *hyperplane* dapat ditemukan dengan mengukur margin *hyperplane* dan mencari titik maksimumnya. Margin adalah jarak antara *hyperplane* dan pola terdekat dari setiap kelas. Pola terdekat disebut vektor dukungan. Garis solid pada Gambar 1 sebelah kanan menunjukkan *hyperplane* terbaik, yang terletak tepat di tengah kedua kelas, sedangkan titik tengah pola yang dilingkari adalah vektor pendukung. Upaya untuk menemukan *hyperplane* terbaik ini merupakan inti dari proses SVM.

2.2. *Malicious Webpage*

Malicious webpage memiliki perbedaan dengan halaman web pada umumnya. Perbedaan pada halaman web tersebut adalah adanya fitur yang digunakan sebagai indikator untuk mendeteksi *malicious webpage*. Fitur tersebut bisa dilihat dari URL dan konten HTML yang di dalamnya juga ada JavaScript. Perbedaan yang secara umum membedakan antara halaman web berbahaya dan yang bukan adalah seperti jumlah karakter konten HTML, jumlah *head tag*, *body tag*, *tag* judul, *tag* elemen *script*, jumlah link yang terdapat pada suatu halaman web, jumlah panjang *string* yang melebihi batas wajar. Selain itu, fitur JavaScript dapat diambil seperti jumlah fungsi *eval ()*, jumlah *string iframe* yang biasanya digunakan untuk tujuan berbahaya [17] dan jumlah fungsi yang digunakan untuk obfuskasi seperti *escape()*. Ciri dari URL yang diambil sebagai fitur URL adalah seperti yang "terlihat berbeda dari biasa", termasuk memiliki URL yang panjang dan terdapat karakter yang sulit dibaca [5], serta memiliki *hosting* yang tidak atau kurang bereputasi [8]. Selain itu, banyak *tag* pada halaman web yang mengarah ke tujuan berbahaya disembunyikan dan biasanya disebut sebagai *hidden tag*. Cara untuk menyembunyikan *tag* tersebut dapat dilakukan dengan JavaScript, jQuery, ataupun CSS [18].

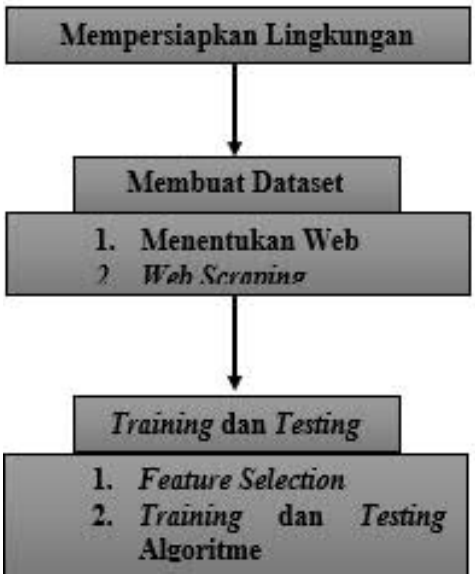
2.3. WEKA

The Waikato Environment for Knowledge Analysis (WEKA) merupakan *tool* yang digunakan untuk penelitian *data mining* karena dapat menyediakan berbagai algoritme *machine learning* dan pemrosesan data. Pengguna *tool* ini dapat dengan mudah mencoba dan membandingkan metode *machine learning* yang berbeda berdasarkan masukkan dataset tertentu [19]. Algoritme yang disediakan WEKA mencakup semua teknik, baik *supervised* ataupun *unsupervised*, seperti algoritme untuk regresi, klasifikasi, *clustering, association rule mining* dan *attribute selection*.

WEKA digunakan untuk proses seleksi fitur dan pelatihan serta pengujian. Semua proses dilakukan dengan menggunakan "*Explorer*" GUI yang dimiliki oleh WEKA. Dengan menggunakan GUI maka dataset yang telah dibuat akan masuk pada panel "*Pre-processing*" dalam bentuk CSV. Selanjutnya, tahap pemilihan fitur akan menggunakan panel "*Select attributes*". Pada panel ini, pemilihan fitur akan menggunakan *CfsSubsetEval Attribute Evaluator* untuk mengetahui fitur mana yang paling berpengaruh pada model *machine learning* yang akan dibuat. Fitur-fitur yang didapat dari penggunaan fitur seleksi nantinya akan digunakan dalam proses pelatihan dan pengujian. Proses pelatihan dan pengujian itu sendiri akan dilakukan di panel "*Classify*".

3. DESAIN

Pada tulisan ini, kami menggunakan metode yang diilustrasikan pada Gambar 2.'



Gambar 2. Metode yang digunakan

3.1. Mempersiapkan Lingkungan

Lingkungan yang disiapkan terdiri dari lingkungan perangkat keras dan perangkat lunak. Perangkat keras yang digunakan merupakan laptop dengan sistem operasi Ubuntu 18.04 LTS, prosesor Intel® Core ™ i3-60060 CPU @ 2.00GHz 1.99 GHz, RAM 4.00 GB, dan *hardisk* 500 GB. Perangkat lunak yang digunakan meliputi Sublime Text utuk membuat kode sumber dan WEKA untuk melakukan *training* dan *testing.*

3.2. Membuat Dataset

3.2.1. Menentukan Web

Web yang akan diambil berasal dari 2 sumber, yakni Alexa Top Sites [20] dan Malware Domain List [21]. Kedua sumber tersebut digunakan sebagai label dari dataset yang akan dibuat, yakni *benign* atau *malicious*. Label *benign* diambil dari Alexa Top Sites dan label *malicious* diambil dari Malware Domain List. Keduanya digunakan berdasarkan rancangan Wang, *et al.* (2017). Alexa Top Sites sendiri merupakan *platform* yang di dalamnya berisi data 500 web teratas yang paling sering dikunjungi di seluruh dunia. Urutan dari 500 web tersebut dapat berubah sewaktu – waktu sesuai dengan intensitas kunjungan terhadap web tersebut setiap harinya. Sedangkan Malware Domain List merupakan *platform* yang di dalamnya berisi daftar URL berbahaya. Daftar tersebut diketahui merupakan data berupa URL yang dapat menjadikan korban terkena *phising*, *malware trojan*, ataupun *ransomware*. Jumlah URL yang didapatkan dari Alexa Top Sites adalah 500. Jumlah URL yang didapatkan dari Malware Domain List berjumlah 1712. Merujuk pada penelitian Wang *et al.* (2017) dilakukan pencarian link atau URL kembali yang ada pada halaman web dari URL Alexa Top Sites, sehingga total URL yang ada saat itu berjumlah 41.134 URL untuk *benign* dan berjumlah 1712 untuk *malicious*.

3.2.2. *Web Scraping*

Web scraping yang dilakukan pada penelitian ini tidak dengan membuat program untuk mendapatkan fitur secara langsung dari URL yang diakses, melainkan dengan mengunduh halaman web dari URL yang ditentukan kemudian baru dilakukan *scraping*. Hal ini dilakukan untuk mengurangi *error* dari dampak koneksi yang buruk ketika sedang melakukan *scraping* dengan mengakses URL secara langsung. Setelah halaman web tersebut diunduh, dilakukan *filtering* terhadap halaman tersebut karena tidak semua halaman yang terunduh dari URL yang didapatkan berisi halaman web HTML. Perlu diketahui juga bahwa berkas yang terunduh tidak hanya memiliki ekstensi HTML, namun dapat berupa ekstensi lain yang tidak menunjukkan berkas tersebut berkas HTML. Setelah dilakukan *filtering* tersebut agar mendapatkan web yang sesuai, didapatkan kembali total 37.535 berkas HTML untuk data *benign* dan 936 berkas HTML untuk data *malicious* yang dapat digunakan. Dari total halaman web yang didapatkan tersebut, data yang akan dimasukkan sebagai dataset mengikuti jumlah yang ada pada rujukan utama, yakni hanya 787 untuk *benign* dan 682 untuk *malicious*.

Setelah mendapatkan semua halaman web yang dibutuhkan, proses *scraping* selanjutnya adalah mengekstrak fitur dari halaman web. Hasil dari ekstraksi fitur halaman web tersebut adalah dataset dokumen berbentuk csv yang akan digunakan pada proses *training* dan *testing* menggunakan WEKA. Tabel 1 berikut merupakan fitur halaman web yang akan diesktrak.

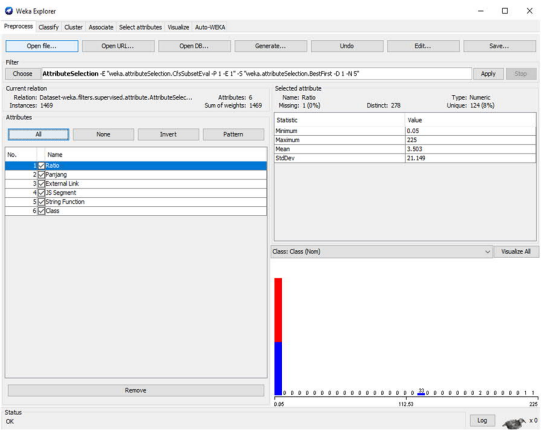
Tabel 1. Fitur Halaman Web

Kategori	Fitur
URL	• Jumlah digit pada URL • Jumlah karakter pada URL • Jumlah simbol pada URL • Rasio spesial karakter pada URL • Tanggal pembuatan URL
HTML	• Jumlah *tag* tersembunyi • Jumlah *link* eksternal • Jumlah *unsymmetric tag* • Jumlah segmen JavaScript • Jumlah *plugin* dan objek ActiveX
Kategori	Fitur
JavaScript	• Jumlah *string* panjang • Jumlah karakter *unicode* • Jumlah pemanggilan fungsi *replace()* • Jumlah fungsi *eval()* dan *exec()* • Jumlah fungsi *string* • Jumlah fungsi obfuskasi, seperti *escape()* dan *unescape()*

3.3. *Training* dan *Testing*

3.3.1. *Feature Selection*

Feature selection dilakukan untuk menghilangkan fitur – fitur yang kurang dibutuhkan menurut WEKA dan harapannya dapat memberikan performa lebih tinggi nantinya berdasarkan dataset yang telah didapatkan. *Feature selection* yang akan digunakan adalah menggunakan *Correlation-based Feature Selection* (CFS) yang mana sudah tersedia di *tool* WEKA sebagai *CfsSubsetEval Attribute Evaluator*. Gambar 3 berikut adalah gambar hasil penggunaan *feature selection*.



Gambar 3. Hasil *Feature Selection*

Berdasarkan *feature selection*, terdapat 5 fitur yang dapat digunakan dan paling berpengaruh

terhadap klasifikasi antara lain, rasio spesial karakter pada URL, panjang URL, jumlah *external link* (HTML), jumlah segmen JS (HTML), dan jumlah fungsi *string* (JavaScript).

3.3.2. *Training* dan *Testing*

Training dan *testing* dilakukan dengan menggunakan algoritme yang telah disebutkan sebelumnya berdasarkan dataset yang telah dibuat. Komposisi yang dibuat untuk data pelatihan dan pengujian adalah 80:20. Tabel 2 menunjukkan hasilnya.

Tabel 2. Performa yang dihasilkan

Algoritme	Performa			
	Akurasi	*Precision*	*Recall*	*F-measure*
Decision Tree	0.921	0.925	0.904	0.914
Naïve Bayes	0.738	0.645	0.963	0.773
SVM	0.802	0.738	0.890	0.807

Berdasarkan Tabel 2 dapat dilihat bahwa rancang bangun model *machine learning* berdasarkan metode Wang, *et al*. (2017) yang menggunakan *decision tree* memberikan nilai performa terbaik dibandingkan dengan menggunakan algoritme lain, seperti *naïve bayes* dan SVM. Nilai kinerja tersebut dapat dilihat dari akurasi, *precision*, dan *f-measure* yang dihasilkan. Meskipun nilai *recall* tertinggi berada pada *naïve bayes*, namun masih jauh dari nilai *precision*, sehingga dibutuhkan nilai *f-measure*. Nilai *f-measure* yang dihasilkan oleh algoritme *naïve bayes* lebih kecil dari metode Wang, *et al.* (2017) yang menggunakan *decision tree*. Nilai tersebut menunjukkan bahwa algoritme selain *decicion tree* masih terdapat kekurangan dalam hal melakukan klasifikasi *malicious webpage*.

4. KESIMPULAN

Kesimpulan dari tulisan ini adalah rancang bangun model *machine learning* terdiri dari tahapan mempersiapkan lingkungan, pembuatan dataset, dan *training* dan *testing* menggunakan algoritme *decision tree*. Model *machine learning* memiliki performa berupa akurasi, *precision*, dan *f-measure* yang dihasilkan adalah 0.921, 0.925, dan 0.914. Penggunaan algoritme *decision tree* memberikan performa terbaik dibandingkan dengan menggunakan algoritme lain seperti *naïve bayes* dan SVM. Performa berupa akurasi, *precision*, dan *f-measure* yang dihasilkan dari algoritme *naïve bayes* adalah 0.738, 0.645, dan 0.773. Performa berupa akurasi, *precision*, dan *f-measure* yang dihasilkan dari algoritme SVM adalah 0.802, 0.738, dan 0.807. Nilai tersebut menunjukkan bahwa algoritme selain *decicion tree* masih terdapat kekurangan dalam hal melakukan klasifikasi *malicious webpage*.

REFERENSI

[1] APJII, "PENETRASI & PROFIL PERILAKU PENGGUNA INTERNET INDONESIA TAHUN 2018," APJII (Asosiasi Penyelenggara Jasa Internet Indonesia), 2019.

[2] S. Yoo and S. Kim, "Two-Phase Malicious Web Page Detection Scheme Using Misuse and Anomaly Detection," *International Journal of Reliable Information and Assurance*, pp. 1-9, 2014.

[3] Securi, "Hacked Website Report 2018," Securi Inc., 2019.

[4] B. Esthe and F. B. Kessler, "Effective Analysis, Characterization, and Detection of Malicious Web Pages," in *International World Wide Web Conference Committee (IW3C2)*, Rio de Jeneiro, 2013.

[5] R. Wang, Y. Zhu, J. Tan and B. Zhou, "Detection of malicious web pages based on hybrid analysis," *Journal of Information Security and Applications*, pp. 68-74, 2017.

[6] H. B. Kazemian and S. Ahmed, "Comparisons of Machine Learning Tecniques for Detecting Malicious Webpages," *Expert Systems with Applications*, pp. 1166-1177, 2015.

[7] Y.-T. Hou, Y. Chang, T. Chen, C.-S. Laih and C.-M. Chen, "Malicious web content detection by machine learning," *Expert Systems with Applications*, pp. 55-60, 2010.

[8] J. Ma, L. Saul, S. Savage and G. Voelker, "Beyond blacklists: learning to detect malicious web sites from suspicious URLs," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge* , Paris, 2009.

[9] D. Patil and Patil, "Detection of Malicious JavaScript Code in Web Pages," *Indian Journal of Science and Technology*, pp. 1-12, 2017.

[10] G. A. Sandag, J. Leopold and V. F. Ong, "Klasifikasi Malicious Websites Menggunakan Algoritma K-NN Berdasarkan Application Layers dan Network Characteristics," *Cogito Smart Journal*, pp. 37-44, 2018.

[11] J. Hurwitz and D. Kirsch, Machine Learning for Dummies, Hoboken: IBM , 2018.

[12] E. Alpaydin, Introduction to Machine Learning, Cambridge: MIT Press, 2004.

[13] C. Danilo and R. Basili, "Decision Tree Algorithm Short Weka Tutorial," 2009. [Online]. Available: http://art.uniroma2.it/basili/MLWM09/002_DecTree_Weka.pdf.

[14] Wesley, "Implementasi Machine Learning pada Sistem Pendeteksi Situs yang Bermuatan Konten Negative," 2019. [Online]. Available: http://repositori.usu.ac.id/handle/123456789/16137.

[15] A. Saleh, "Implementasi Metode Klasifikasi Naive Bayes Dalam Memprediksi Besarnya Pengunaan Listik Rumah Tangga," *Citec Journal*, pp. 207-217, 2015.

[16] A. S. Nugroho, A. B. Witarto and D. Handoko, "Support Vector Machine dan Aplikasinya dalam Bionformatika," 2003.

[17] P. S. and C. Thomas, "A Static Approach to Detect Drive-by-download Attacks on Webpages," in *International Conference on Control Communication and Computing (ICCC)*, 2013.

[18] J. H. Nezhad, M. V. Jahan, M. Tayarani-N and Z. Sadrnezhad, "Analyzing New Features of Infected Web Content in Detection of Malicious Webpages," *Internation Journal of Information Security*, pp. 161-180, 2017.

[19] M. Hall, E. Frank, G. Holmes and B. Pfahringer, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, pp. 10-18, 2009.

[20] Alexa Internet, Inc, "Top Sites in Indonesia," 2019. [Online]. Available: https://www.alexa.com/topsites/countries/ID.

[21] Malware Domain List, 2019. [Online]. Available: http://www.malwaredomainlist.com/mdlcsv.php.