Implementasi Algoritma *Supervised Learning* untuk Identifikasi *Malware* Berbasis Python *Module*

1st Muhammad Sagi Dimarzio
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
sagidimarzio@student.telkomuniversity.ac.id

2nd Rohmat Tulloh
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
rohmatth@telkomuniversity.ac.id

3rd Muhammad Iqbal
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
miqbal@telkomuniversity.ac.id

***Abstrak* —*Malware* (*Malicious Software*) merupakan salah satu ancaman pada internet yang selalu berkembang dengan cepat, beragam, dan semakin kompleks. Antivirus dikenal sebagai mitigasi utama malware, namun pada zaman sekarang diperlukan bantuan tenaga keamanan siber profesional. Tetapi, sumber daya manusia di bidang keamanan siber yang secara spesifik menekuni analisis malware tentunya juga terbatas. Berdasarkan permasalahan di atas, salah satu cara untuk memitigasi masalah tersebut adalah dengan menggunakan teknologi heuristic detection. Heuristic Detection dapat dicapai melalui metode Supervised Learning pada salah satu teknologi kecerdasan buatan, yaitu Machine Learning. Proyek Akhir ini akan fokus pada implementasi model algoritma Supervised Learning menggunakan bahasa pemrograman Python untuk mendeteksi malware berdasarkan perilaku dan atribut software yang akan diidentifikasi.**

Hasil pengujian membuktikan nilai akurasi sebesar 93,3% dan nilai presisi sebesar 90,9%.

Kata kunci* —*cybersecurity, malware, python, supervised learning

I. PENDAHULUAN

Internet telah menjadi bagian besar hidup manusia dalam dua dekade ke belakang. Transformasi digital pada sebagian besar produk dan layanan jasa untuk membantu kemudahan akses konsumen menjadi penyebab utama penggunaan internet kian menjadi krusial. Namun, dibalik kecanggihan internet yang menarik masyarakat, ada sisi bahaya yang dimiliki internet, yaitu *cyber threat* atau *threat* (ancaman).

Menurut Symantec Inc., kejahatan siber pada April 2012 dikalkulasikan menyebabkan kerugian senilai US$ 114 Miliar tiap tahunnya [1]. Pada akhir Januari 2020, total *malware* baru yang terdeteksi sejumlah 661 Juta (Statista, 2021). Mengapa kejahatan siber seperti *hacking*/meretas, *malware*, dan *phishing* sangat marak? Tanpa kita sadari, kejahatan siber lebih murah, tidak melibatkan aktivitas fisik yang berat, dan tidak terlalu berisiko. Terutama penyebaran *malware* yang sangat mudah terjadi apabila gawai kita terkoneksi ke internet.

Latar belakang permasalahan tersebut menginspirasi penulis untuk meneliti cara memitigasi ancaman malware. Penelitian ini akan fokus kepada pembuatan program yang berfungsi untuk mengidentifikasi keabsahan file berekstensi .exe yang telah diunduh melalui internet. Program akan mampu membedakan file yang sah dengan yang *malware*. Pembuatannya akan menggunakan bahasa pemrograman Python dan memanfaatkan teknologi *machine learning* dengan algoritma *supervised learning*

II. KAJIAN TEORI

A. *Malware*

Malicious Software atau yang lebih dikenal sebagai *malware* merupakan perangkat lunak yang secara eksplisit didesain untuk melakukan tindakan melanggar hukum. *Malware* diciptakan dengan maksud tertentu yaitu melakukan aktifitas berbahaya yang berdampak sangat merugikan bagi para korbannya, antara lain seperti penyadapan perangkat, pencurian informasi pribadi, hingga kasus perusakan sistem yang dilakukan oleh penyerang (*threat actor*) terhadap asset digital korban dengan berbagai alasan [2].

B. *Python*

Python adalah bahasa pemrograman interpretatif yang dapat digunakan di berbagai platform dengan perancangan yang berfokus pada keterbacaan kode dan merupakan salah satu bahasa populer yang seringkali digunakan pada web, program otomasi, Data Science, Artificial Intelligence (AI), dan Internet of Things (IoT).

C. *Artificial Intelligence*

Artificial Intelligence (AI) adalah simulasi dari kecerdasan yang dimiliki oleh manusia yang dimodelkan ke dalam mesin dan diprogram agar bisa berpikir seperti halnya manusia. Dengan kata lain AI merupakan sistem komputer yang bisa melakukan pekerjaan-pekerjaan yang umumnya memerlukan tenaga manusia atau kecerdasan manusia untuk menyelesaikan pekerjaan tersebut.

D. *Machine Learning*

Machine Learning (ML) adalah mesin yang dikembangkan untuk bisa belajar dengan sendirinya tanpa arahan dari penggunanya. Pembelajaran mesin

dikembangkan berdasarkan disiplin ilmu lainnya seperti statistika, matematika dan data mining sehingga mesin dapat belajar dengan menganalisa data tanpa perlu di program ulang atau diperintah [3].

Dalam hal ini machine learning memiliki kemampuan untuk memperoleh data yang ada dengan perintahnya sendiri. ML juga dapat mempelajari data yang ada dan data yang ia peroleh sehingga bisa melakukan tugas tertentu. Tugas yang dapat dilakukan oleh ML pun sangat beragam, tergantung dari apa yang ia pelajari.

Ada 2 teknik yang digunakan pada machine learning, yaitu:

1. *Supervised Learning*

Teknik supervised learning merupakan teknik yang bisa diterapkan pada pembelajaran mesin yang bisa menerima informasi yang sudah ada pada data dengan memberikan label tertentu. Diharapkan teknik ini bisa memberikan target terhadap output yang dilakukan dengan membandingkan pengalaman belajar di masa lalu.

2. *Unsupervised Learning*

Teknik unsupervised learning merupakan teknik yang bisa diterapkan pada machine learning yang digunakan pada data yang tidak memiliki informasi yang bisa diterapkan secara langsung. Diharapkan teknik ini dapat membantu menemukan struktur atau pola tersembunyi pada data yang tidak memiliki label.

Sedikit berbeda dengan *supervised learning*, proses learning tidak memiliki data apapun yang akan dijadikan acuan sebelumnya sebagai keluaran. Sehingga, hasil dari *learning* bukan sesuatu yang menjadi output logika, melainkan pengelompokan dari data set yang ada.

E. Random Forest

Random Forest (RF) merupakan algoritma *machine learning* yang berdasar pada pohon keputusan atau *decision tree* dan *bagging*. Algoritma RF pada umumnya sering digunakan untuk regresi dan klasifikasi masalah.RF memanfaatkan *ensemble learning*, yang menggabungkan beberapa *classifiers* untuk menemukan solusi terbaik. Random forest merupakan kombinasi dari masing – masing pohon (*tree*) dari model Decision Tree yang baik, dan kemudian dikombinasikan ke dalam satu model.
Penggunaan tree yang semakin banyak akan mempengaruhi akurasi yang akan didapatkan menjadi lebih baik. Penentuan klasifikasi dengan random forest diambil berdasarkan hasil *voting* dari pohon yang terbentuk. Pemenang dari pohon yang terbentuk ditentukan berdasarkan pilihan terbanyak. Pembangunan pohon (*tree*) pada random forest sampai dengan mencapai ukuran maksimum dari pohon data.

F. Decision Tree

Decision Tree (DC) adalah algoritma yang di mana prediksi yang dihasilkan berdasarkan pada beberapa peraturan yang dipisah-pisahkan. Peraturan tersebut diwakili oleh *nodes*, peraturan dipisah berdasarkan cabang-cabang yang ada, dan prediksi akhir ditentukan oleh daun-daun dari pohon tersebut. Konsepnya adalah dengan cara menyajikan algoritma dengan pernyataan bersyarat, yang meliputi cabang untuk mewakili langkah-langkah pengambilan keputusan yang dapat mengarah pada hasil yang menguntungkan. Setiap cabang mewakili hasil, sedangkan jalur dari daun ke akar mewakili aturan untuk klasifikasi.

Konsep dari pohon keputusan adalah mengubah data menjadi decision tree dan aturan-aturan keputusan. Manfaat utama dari penggunaan decision tree adalah kemampuannya untuk menjabarkan proses pengambilan keputusan yang kompleks menjadi lebih simple, sehingga pengambil keputusan akan lebih menginterpretasikan solusi dari permasalahan..

G. Gradient Boosting

Gradient Boosting (GB) adalah salah satu teknik atau algoritma ML yang digunakan untuk regresi, prediksi, dan klasifikasi. Model prediksi yang dihasilkan dari penggunaan algoritma GB berbentuk form dari gabungan model prediksi yang lemah. Cara kerja algoritma gradient boosting adalah membangun satu pohon untuk menyesuaikan data, lalu pohon berikutnya dibangun untuk mengurangi kesalahan data dalam prosesnya.

H. Gaussian Naïve Bayes

Gaussian Naive Bayes adalah varian dari Naive Bayes yang mengikuti distribusi Gaussian dan memanfaatkan data bersifat yang kontinu. Model ini merupakan algoritma klasifikasi dengan probabilitas berdasarkan penerapan teorema Bayes dengan asumsi bahwa tiap kolom data memiliki independensi yang kuat. Model ini memprediksi peluang di masa depan berdasarkan pengalaman atau hasil data di masa sebelumnya, sehingga cocok untuk identifikasi probabilitas suatu nilai.

I. Adaboost

Adaboost adalah algoritma dalam *machine learning* yang digunakan sebagai *Ensemble Method*. *Ensemble Method* adalah metode pada *machine learning* yang menggabungkan beberapa model dasar dengan tujuan menghasilkan model dengan kemampuan prediksi yang optimal. Hal tersebut dilakukan dengan mengkombinasikan sekumpulan fungsi pengklasifikasi yang lemah untuk membentuk sebuah pengklasifikasi yang lebih kuat. Untuk menentukan kekuatan klasifikasinya, model menggunakan rangkaian pohon untuk melatih keputusan-keputusan model.

J. Linear Regression

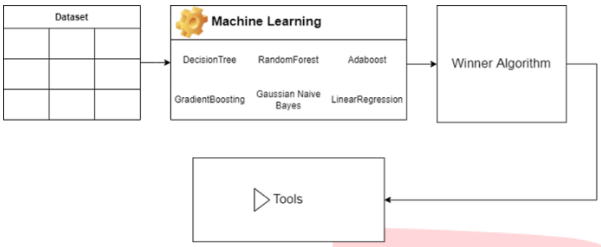
Linear Regression adalah algoritma model yang menentukan nilai prediksi berdasarkan variabel bersifat independen. Perhitungan model ini dilakukan secara statistik untuk memprediksi data yang nilainya kontinu, sehingga mampu memprediksi nilai berikutnya. Nilai yang diprediksi akan dikorelasikan dengan variable-variabel yang terdapat pada nilai sebelumnya. Penggunaan model linear regression lebih cocok dengan data yang variabelnya independen dan tidak saling terkorelasi antar satu sama lain untuk mencapai prediksi terbaiknya.

III. HASIL DAN PEMBAHASAN

A. Block Diagram Sistem

Adapun model sistem monitoring yang telah dibuat dapat dilihat pada Gambar 3.1.

Sistem identifikasi malware menggunakan 6 model algoritma, kemudian hasil komparasi tingkat akurasi masing-masing model akan menentukan algoritma akhir yang akan digunakan sebagai deteksi malware. Model algoritma yang akan digunakan antara lain: Decision Tree, Random Forest, Adaboost, Gradient Boosting, Gaussian Naïve Bayes, dan Linear Regression.



GAMBAR 1
SISTEM PERANCANGAN PROGRAM IDENTIFIKASI MALWARE

B. Proses Perancangan Proyek Akhir

Proses perancangan system identifikasi malware menggunakan tahapan sebagai berikut:

1. Pembuatan script Python.

Langkah awal adalah membangun script untuk pemrosesan machine learning menggunakan bahasa pemrograman Python.

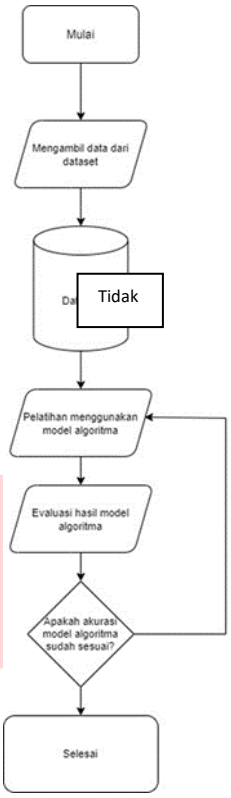
2. Melatih mesin menggunakan model algoritma.

Selanjutnya melakukan pelatihan terhadap mesin yang dirancang menggunakan model algoritma Decision Tree, Random Forest, Adaboost, Gradient Boosting, Gaussian Naïve Bayes, dan Linear Regression.

3. Mengevaluasi akurasi mesin.

Apabila mesin sudah mencapai akurasi yang optimal, maka mesin dinyatakan berhasil dan layak digunakan sebagai prototype.

Diagram alir dalam tahapan perancangan dapat dilihat pada Gambar 2.



GAMBAR 2
DIAGRAM BLOK PERENCANAAN

C. Set Data

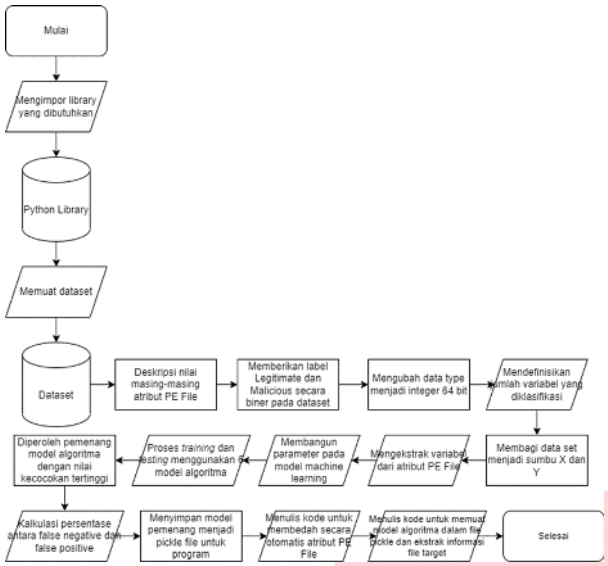
Set Data atau dataset yang akan digunakan adalah file dengan ekstensi *Portable Executable* (PE) yang telah terbukti sebagai *malware*. Data ini merupakan kumpulan *malware* yang layak digunakan sebagai set data untuk percobaan identifikasi pada penelitian ini. Selain itu, ekstensi file PE juga mengandung beberapa informasi dari file data tersebut seperti *size of code*, *size of optional header*, *address of entry point*, *base of code*, dan lain sebagainya.

Atribut – atribut tersebut dapat membantu untuk mengembangkan konteks yang lebih luas dan akurat mengenai file atau data tersebut.

D. Alur Pembangunan Aplikasi

Pada penelitian ini akan merancang program menggunakan bahasa pemrograman Python untuk identifikasi *malware* menggunakan teknologi *Machine Learning* dengan implementasi algoritma *Supervised Learning*.

Proses pembangunan program akan dilakukan pada Google Colab, yaitu sebuah aplikasi berbasis cloud untuk menulis dan mengeksekusi kode yang mendukung lebih dari 40 bahasa pemrograman.



IV. KESIMPULAN

Berdasarkan hasil perancangan, pengujian, dan analisis yang telah dilakukan maka diperoleh beberapa kesimpulan sebagai berikut:

A. Pendekatan Supervised Learning terbukti cocok untuk membangun machine learning dengan tujuan klasifikasi data.
B. Model algoritma Random Forest terbukti cocok dengan data set yang digunakan, berdasarkan hasil training dan testing dengan nilai 99,43%.
C. Namun, setelah pengujian didapat bahwa nilai akurasi sebesar 93,33% dan nilai presisi sebesar 90,09% menunjukkan bahwa hasil pengujian tools belum mencapai keluaran yang optimal.
D. Dapat mempercepat hasil analisis statis PE File ketika diolah dengan machine learning..

REFERENSI

[1] J. Jang-Jaccard and S. Nepal, “A survey of emerging threats in cybersecurity,” J. Comput. Syst. Sci., vol. 80, no. 5, pp. 973–993, 2014, doi: 10.1016/j.jcss.2014.02.005.
[2] T. A. Cahyanto, V. Wahanggara, and D. Ramadana, “Analisis dan Deteksi Malware Menggunakan Metode Malware Analisis Dinamis dan Malware Analisis Statis,” Justindo, J. Sist. Teknol. Inf. Indones., vol. 2, no. 1, pp. 19–30, 2017, [Online]. Available: http://jurnal.unmuhjember.ac.id/index.php/JUSTINDO/article/view/1037.
[3] E. S. Lamdompak Sistem Komputer and F. Ilmu Komputer, “Klasifikasi Malware Trojan Ransomware Dengan Algoritma Support Vector Machine (SVM),” vol. 2, no. 1, pp. 122–127, 2016, [Online]. Available: http://ars.ilkom.unsri.ac.id.
[4] G. A. Sandag, J. Leopold, and V. F. Ong, “Klasifikasi Malicious Websites Menggunakan Algoritma K-NN Berdasarkan Application Layers dan Network Characteristics,” CogITo Smart J., vol. 4, no. 1, p. 37, 2018, doi: 10.31154/cogito.v4i1.100.37-45.
[5] L. Wen and H. Yu, “An Android malware detection system based on machine learning,” AIP Conf. Proc., vol. 1864, no. August 2017, 2017, doi: 10.1063/1.4992953.
[6] A. Bijalwan, “Botnet Forensic Analysis Using Machine Learning,” Secur. Commun. Networks, vol. 2020, 2020, doi: 10.1155/2020/9302318.
[7] N. Bhodia, P. Prajapati, F. Di Troia, and M. Stamp, “Transfer learning for image-based malware classification,” ICISSP 2019 - Proc. 5th Int. Conf. Inf. Syst. Secur. Priv., pp. 719–726, 2019, doi: 10.5220/0007701407190726.
[8] Y. M. Cho and H. Y. Kwon, “API Call Time Interval을 활용한 머신러닝 기반의 악성코드 탐지,” vol. 30, no. 1, pp. 51–58, 2020.