Prediksi Jaringan TOR dan VPN menggunakan Algoritma *K-Nearest Neighbour* pada Trafik Darknet

Aay Ramdan
Informatika
Universitas Siliwangi
Tasikmalaya, Indonesia
aayramdan@gmail.com

Nur Widyasono
Informatika
Universitas Siliwangi
Tasikmalaya, Indonesia
Nur.widyasono@unsil.ac.id

Husni Mubarok
Informatika
Universitas Siliwangi
Tasikmalya
kbm.hm@unsil.ac.id

Abstract—The process of network forensics for analyzing malware has been carried out by previous researchers by applying manual methods including the Behavior Anomaly method on network traffic capture files. Such network forensics requires a longer process and is not accurate with the desired results. The rapid development of artificial intelligence in every field of technology can provide opportunities for the field of malware analysis and digital forensics to be able to carry out the analysis process more quickly and precisely, especially the use of Machine Learning. Darknet traffic is an internet network in which there are various threats of cyber crime. Many researches on malware analysis, especially darknet traffic classification using machine learning algorithms, have been carried out, but the results obtained are in the form of performance measurements on each machine learning algorithm for the malware analysis process without any dataset updates or implementation in an application. Dataset updates are very necessary so that malware analysis can identify the latest malware developments and implementation is carried out in order to know the performance of an applied algorithm, because this research will discuss the process of analyzing malware threats on darknet traffic using machine learning algorithms, namely K- Nearest Neighbor to predict a malware attack threat with the CICDarknet 2020 dataset. The results of the dataset performance measurement using KNN have an accuracy value of 96.17% by applying feature selection with information gain.

Keywords—Darknet, KNN, Malware, Network, Forensic

I. PENDAHULUAN

Keamanan informasi merupakan suatu hal penting dalam era digital yang mengintegrasikan semua aspek ke dalam internet. Beberapa aspek yang harus dijaga dalam sebuah informasi yaitu *Confidentiality*, *Integrity*, *Avaibility*, *Authentication*, *Authorization* dan *Non Repudation* untuk memastika bahwa informasi tersebut tidak terserang oleh pelaku kejahatan internet [1]. Suatu informasi dapat dipastikan aman sangatlah sulit dikarenakan banyaknya penjahat internet yang berusaha untuk menyerang suatu sistem [2].

Kejahatan di dunia *cyber* telah merambat ke dunia bisnis, pemerintah bahkan individu, sehingg hal tersebut dapat dijadikan perhatian agar keamanan dalam dunia *cyber* ditingkatkan guna meminimalisir kejahatan yang terjadi [3].

Analisa dinamis dari sebuah trafik di dalam internet diperlukan untuk mengetahui ancaman serangan malware dengan memerhatikan perilakunya di dalam jaringan internet [4]. Data trafik internet yang berjumlah banyak membuat proses analisa dinamis secara manual sulit untuk dilakukan, sehingga perlu adanya sebuah algoritma *Machine Learning* yang dapat memeriksa banyak trafik sekaligus.

Proses *network forensic* memerlukan analisis yang teliti dan membutuhkan waktu yang lama untuk menemukan informasi dalam sebuah data *evidence* hasil akuisisi [5]. Praktik dalam proses digital forensik notabenenya dilakukan oleh

seorang yang awam akan teknologi yang mendalam terutama tentang digital forensik [6]. *Investigator* dituntut untuk mampu melakukan analisa terhadap suatu ancaman *malware* yang terus berkembang [7]. Salah satu cara untuk mengatasi masalah tersebut adalah penerapan machine learning pada proses analisis data evidence.

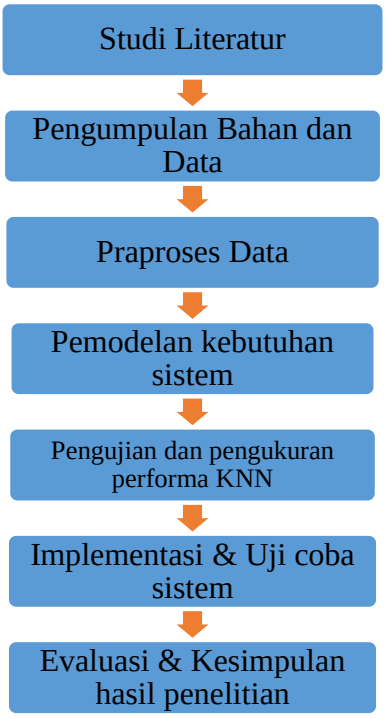
Paket – paket data yang ditargetkan pada trafik darknet dianggap mencurigakan. Paket – paket ini sering dibuat oleh malware atau penyerang saat mencari target potensial berikutnya [8]. Trafik *darknet* menjadi hal yang perlu diperhatikan untuk mengetahui secara dini akan adanya ancaman serangan malware.

Penelitian serupa terkait penelitian ini yaitu tentang analisis ancaman malware dengan menggunakan dataset SURFnet yang didalamnya terdapat trafik darknet dengan menggunakan algoritma *Machine Learning* [9]. Hasil penelitian Kumar menunjukkan bahwa dataset SURFnet masih perlu dikembangkan terkait dengan fitur dataset yang masih sedikit dan hanya memiliki dua label, hal ini diperlukan dikarenakan ancaman malware juga terus ikut berkembang.

Latar belakang tersebut menjadi dasar diperlukannya sebuah proses digital forensik yang didalamnya terdapat penerapan machine learning untuk proses analisis data *evidence* hasil akuisisi. Ilmu yang menggabungkan dua bidang ilmu yang berbeda yaitu machine learning dan forensik ini kemudian disebut *Intelligence Forensic*. *Intelligence forensic* hasil penelitian akan menyajikan proses analisis data *evidence* lebih *user friendly* dan dapat dipahami dengan mudah oleh para investigator yang awam tentang analisis data forensik.

II. METODOLOGI

Tahapan penelitian digambarkan secara lengkap dengan menggunakan *flow chart* dengan menggunakan metode penelitian kuantitatif. *Flow chart* atau diagram alur digunakan untuk memudahkan penyampaian informasi terkait langkah- langkah yang akan dilakukan dalam penelitian ini. Kebaruan yang ditargetkan dari penelitian yang diusulkan adalah prediksi ancaman serangan *malware* dengan menggunakan dataset terbaru yaitu CICDarknet 2020 yang memiliki label lebih detail dari dataset Surfnet dengan menggunakan algoritma *K-Nearest Neighbour*. Tahapan penelitian keseluruhan disajikan pada gambar 1.



Gambar 1. Tahapan Penelitian

A. Studi Literatur

Melakukan kajian pustaka yang menunjang penelitian diantaranya terkait analisis *malware*, *machine learning*¸ algoritma KNN dan proses digital forensik. Literatur yang dikaji diperoleh dari jurnal – jurnal dan buku yang sumbernya terpercaya.

B. Pengumpulan Bahan dan Data

Data yang dibutuhkan untuk penelitian ini adalah adalah data primer hasil tangkapan trafik jaringan sebagai data uji dan data sekunder dari hasil penelitian Canadian Institut for Cybersecurity yaitu dataset CICDarknet 2020 yang di dalamnya berisi data data hasil tangkapan jaringan nyata yang memiliki trafik *darknet* diantaranya TOR dan VPN sebagai data latih. Data primer diharuskan memiliki kualitas data yang baik, oleh karena itu diperlukan proses ekstraksi fitur, validasi, integrasi dan transformasi serta reduksi ukuran data dan diskretisasi data agar sesuai dengan data latih yang telah disiapkan [10].

C. Pemodelan Kebutuhan Sistem

Pemodelan dibuat dengan menggunakan jupyter-lab, data hasil tangkapan akan dilakukan praproses data terlebih dahulu untuk memastikan data memiliki kualitas yang baik. Praproses data dilakukan dengan tahapan ekstraksi fitur dengan menggunakan CICFlowmeter, kemudian akan dilakukan validasi, integrasi dan transformasi terhadap data hasil tangkapan. Data latih dan data uji akan diunggah ke dalam model kemudian dilakukan seleksi fitur untuk mengurangi waktu proses klasifikasi.

D. Pengujian dan Pengukuran Performa KNN

Hasil data yang telah diseleksi akan dilakukan pelatihan dan pengujian dengan menggunakan algoritma KNN untuk melihat nilai performa dari dataset yang

digunakan. *Hypertuning parameter* atau pengujian nilai k secara berulang adalah salah satu teknik yang digunakan untuk melakukan pelatihan dataset untuk mendapatkan parameter k terbaik. K terbaik akan menjadi tolak ukur pengujian, percobaan dilakukan sebanyak dua kali, yaitu pelatihan dan pengujian dengan fitur lengkap serta dengan fitur terpilih.

E. Implementasi dan Uji Coba Sistem

Implementasi dilakukan dengan pembuatan sebuah sistem yang dapat memprediksi ancaman serangan yang terdapat dalam file PCAP dengan menggunakan algoritma KNN. Bahasa pemrograman yang digunakan adalah python yang memiliki modul untuk proses klasifikasi dan prediksi. Uji coba sistem akan dilakukan dengan menilai seberapa lama pemroresan klasifikasi dan prediksi data uji yang diunggah ke sistem.

F. Pengumpulan Bahan dan Data

Evaluasi dilaksanakan dengan menggunakan metode *Confussion Matrix* yang biasanya digunakan dalam evaluasi model pada kasus klasifikasi untuk menghitung tingkat akurasi, presisi, *recall* dan *f1-score*.

Hasil penelian akan disimpulkan dengan memerhatikan hasil pengukuran algoritma KKN dengan fitur lengkap dan fitur terpilih setelah dilakukan seleksi fitur.

III. HASIL DAN PEMBAHASAN

A. Data Trafik Darknet

Data penelitian menggunakan data primer untuk digunakan sebagai data yang akan diprediksi dengan menggunakan dataset CICDarknet 2020 yaitu hasil tangkapan jaringan dengan menggunakan wireshark yang dilakukan dengan melakukan percobaan akses terhadap situs darknet dengan menggunakan *web browser* TOR dan VPN serta data sekunder sebagai data latih yaitu dataset CICDarknet 2020 yang diperoleh dari projek Canadian Institute for Cybersecurity.

B. Praposes Data

Praproses data adalah salah satu proses yang perlu dilakukan untuk mendapatkan data dengan kualitas yang baik dengan cara diantaranya validasi, integrasi dan transformasi [11]. Praproses data meliputi pemeriksaan dan pembuangan data yang inkonsisten, data ganda, data yang perlu diperbaiki dan penambahan data sesuai dengan yang dibutuhkan. Dataset CICDarknet memiliki beberapa nama fitur yang berbeda dengan data hasil ekstraksi jaringan. Penamaan ulang diperlukan untuk menyamakan nama setiap fitur pada dataset untuk memudahkan dalam proses pengolahan data selanjutnya. Nama setiap fitur akan dijadikan header untuk proses pengolahan data selanjutnya.

1) Pembersihan Data

Pembersihan data merupakan salah satu pra-proses data untuk menghilangkan data yang kosong dan tidak lengkap. Data yang rusak dapat disebabkan karena adanya kesalahan pada saat transfer data, kesalahan yang disebabkan oleh sistem ataupun permasalahan lainnya. Hal tersebut akan mengakibatkan kesulitan dalam proses analisis yang dilakukan. Data tersebut perlu dihilangkan atau diganti dengan nilai konstan atau nilai tengah dari data yang sekolom. Data yang rusak

dapat dilihat dengan menggunakan fungsi *describe()* dari *library* pandas, fungsi ini dapat menunjukan jumlah data, rata – rata, deviasi, data terendah, data 25%, 50%, 75% dan data tertinggi dari setiap fitur dalam dataset yang digunakan. Fungsi *describe()* hanya menunjukan atribut numerik dari dataset, selain itu pula fungsi ini mampu mengetahui fitur apa saja yang tidak memiliki data yang diperlukan dalam proses analisis. Fitur – fitur tersebut tidak akan mempengerahui proses pelatihan dan pengujian data, oleh karena itu untuk mengefesienkan proses analisis, fitur – fitur yang tidak terpakai dapat dihapus dengan fungsi *drop().*

Fitur yang tersisa diperiksa kembali untuk memastikan tidak ada nilai yang rusak atau tidak sesuai dengan data yang dibutuhkan. Data yang memiliki nilai Nan dan Infinity dapat menghambat proses analisis, oleh karena itu data yang memiliki nilai *NaN* dan *Infinity* harus dicek terlebih dahulu keberadaannya dalam dataset. Fungsi yang digunakan untuk mengecek nilai *NaN* dan *Infinity* adalah fungsi *isna().any()* dari Pandas. Fungsi ini dapat menentukan dan mengecek seluruh fitur dan menunjukan hasil *True* atau *False*. Hasil *True* menunjukan data memiliki nilai *Nan* atau *Infinity* dan *False* menunjukan data normal dan tidak perlu dilakukan proses selanjutnya.

Fitur flow_bytes menunjukan nilai true yang menunjukkan bahwa fitur tersebut memiliki nilai *NaN* dan *Infinity* sehingga menyebabkan tipe data menjadi ambigu, oleh sebab itu data tersebut akan dihapus untuk memudahkan proses analisis data selanjutnya. Dimensi data yang telah dilakukan pembersihan data akan mengalami reduksi sehingga pada proses ini jumlah fitur menjadi 80 fitur dan jumlah data dikurangi data yang memiliki *NaN* dan *infinity* menjadi 141.481 data seperti yang terlihat pada gambar berikut.

```
print('Dimensi Data : ',df.shape)

Dimensi Data :  (141481, 80)
```

Gambar 2. Dimensi Data Setelah Pembersihan Data

2) Transformasi Data

Transformasi data merupakan salah satu hal yang penting dalam praproses data untuk digunakan dalam salah satunya proses normalisasi data agar nilai data tidak memiliki rentang yang jauh dengan data lainnya dengan menggunakan skala tertentu [12]. Teknik normalisasi yang digunakan adalah *Z-Score Normalization* yang merupakan metode normalisasi dengan menggunakan mean atau nilai rata – rata dan standar deviasi dari data yang akan dinormalisasi seperti terlihat pada persamaan 1:

$$\text{Normalisasi}(xi) = \frac{x_i - mean(x)}{stdev(x)} \quad (1)$$

Normalisasi atribut numerik ini dimaksudkan agar nilai yang ada pada dataset menjadi lebih kecil dan tidak memiliki rentang nilai yang berjauhan tanpa mengubah esensi nilai dari setiap data. Perhitungan manual normalisasi dapat dilihat pada penjelasa berikut. Data yang digunakan adalah dataset CICDarknet 2020 yang telah dilakukan proses pembersihan data yang dapat dilihat pada tabel 1 berikut.

Tabel 1. Sampel Data Normalisasi Atribut Numerik

No	src_port	dst_port	flow_duration	total_fwd_packet	...	idle_min
1	57158	443	229	1	...	0.0
2	57159	443	407	1	...	0.0
3	57160	443	431	1	...	0.0
4	49134	443	359	1	...	0.0
...	...	...	...	...	...	...
141530	11666	60245	119990044	5995	...	0.0

Implementasi untuk proses normalisasi dapat dilakukan dengan menggunakan fungsi *StandardScaler()* yang disediakan *library sklearn*. Fitur dengan atribut numerik dipisah dengan fitur yang memiliki atribut kategorik karena normalisasi hanya dapat dilakukan pada data yang berupa angka atau numerik. Proses normalisasi dengan fungsi *StandardScaler()* ditunjukan pada gambar 3 di bawah.

```
df.head()
```

	src_port	dst_port	protocol_protocol	flow_duration	total_fwd_packet	total_bwd_packet	total_length_fwd_packet
0	0.977997	-0.796097	-0.801148	-0.546508	-0.063838	-0.044949	-0.034644
1	0.978049	-0.796097	-0.801148	-0.546503	-0.063838	-0.044949	-0.034644
2	0.978101	-0.796097	-0.801148	-0.546502	-0.063838	-0.044949	-0.034644
3	0.558265	-0.796097	-0.801148	-0.546504	-0.063838	-0.044949	-0.034644
4	-0.196928	0.053620	-0.801148	-0.263585	0.184194	0.071741	-0.014801

Gambar 3. Normalisasi dataset CIC Darknet 2020

C. Seleksi Fitur

Seleksi fitur merupakan salah satu proses dalam machine learning untuk menentukan fitur mana saja yang memiliki pengaruh yang tinggi terhadap fitur label atau keputusan dalam dataset. Teknik yang dilakukan dalam seleksi fitur salah satunya adalah information gain. Information gain dilakukan dengan cara melakukan pendekatan terhadap penyaringan *(filtered based)* dari dataset [13]. Proses penghitungan information gain dari setiap atribut dilakukan dengan menghitung entropi masing – masing atribut terlebih dahulu dengan menggunakan persamaan 2 berikut:

$$\text{Entropi (S)} = \sum_{j=1}^{k} - p_j \, log_2 p_j \qquad (2)$$

Dimana:

- S adalah himpunan dari dataset kasus
- k adalah banyaknya bagian atau partisi dari S
- p_j adalah peluang yang didapatkan dari jumlah dibagi total kasus.

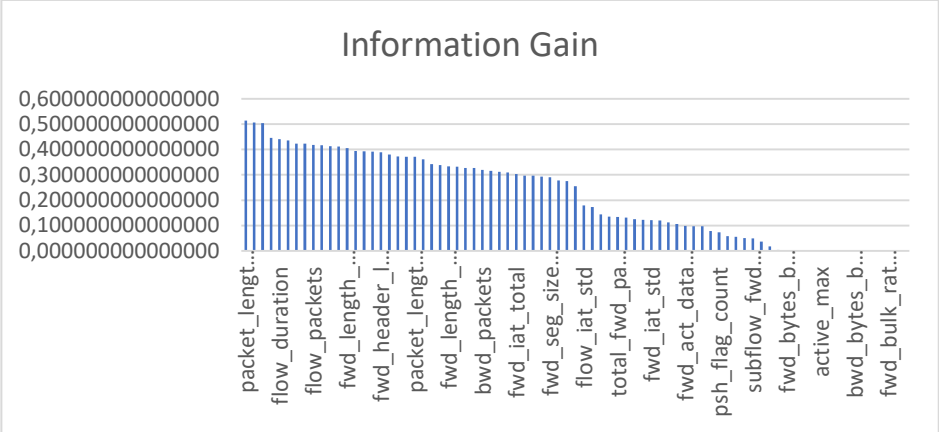
Hasil dari perhitungan entropi setiap atribut dijadikan acuan untuk untuk melakukan pemilihan atribut yang dilakukan dengan nilai information gain terbesar. Nilai information gain dapat dihitung dengan persamaan 3 berikut:

$$\text{Gain (A)} = \text{Entropi (S)} - \sum_{i=1}^{k} \frac{|S_i|}{|S|} x \; Entropi(S_i) \quad (3)$$

Dimana:

- S adalah ruang data sample yang digunakan untuk pelatihan.
- A adalah atribut yang akan dihitung
- |Si| adalah jumlah sampel untuk nilai V
- |S| adalah jumlah seluruh sampel data
- Entropi (Si) adalah entropi untuk sampel - sampel yang memiliki nilai i.

Fungsi yang digunakan untuk memperoleh nilai information gain di sklearn adalah fungsi mutual_info_classif(). Hasil dari proses penghitungan information gain dengan fungsi mutual_info_classif() dapat dilihat pada gambar 4 berikut.



Gambar 4. Grafik nilai information gain setiap fitur

Pemilihan atribut pada penelitian ini ditetapkan fitur dengan bobot lebih dari 0,3 untuk nilai information gain pada setiap fitur dalam dataset CICDarknet 2020, sehingga fitur yang tidak memenuhi syarat akan dihapus. Tabel 2 menunjukkan 33 fitur hasil seleksi dari proses seleksi fitur dengan menggunakan information gain.

Tabel 2. Seleksi fitur dataset CICDarknet 2020

No	Fitur	Hasil Information Gain
1	packet_length_mean	0,513890007765810
2	average_size	0,506159720376371
3	packet_length_max	0,504644183278891
4	flow_iat_max	0,446413493311978
5	flow_duration	0,441487334836825
6	total_length_fwd_packet	0,435652327502528
7	dst_port	0,423489353998881
8	flow_iat_min	0,422712616306851
9	flow_packets	0,418715975806957
10	flow_iat_mean	0,416587739567886
11	fwd_packets	0,412692983617843
12	flow_bytes	0,411197470994761
13	fwd_length_max	0,406100854567985
14	fwd_initwin_bytes	0,393500754077179
15	fwd_segment_avg	0,392711997969578
16	fwd_length_mean	0,391162772511376
17	fwd_header_length	0,389672473308522
18	src_port	0,380626619088048
19	packet_length_variance	0,372490837969473
20	idle_max	0,371579979975883
21	packet_length_std	0,371482567576949

22	idle_mean	0,360869020015187
23	bwd_length_max	0,342349376202303
24	idle_min	0,338658113738450
25	fwd_length_min	0,333002015301377
26	packet_length_min	0,331738496779290
27	bwd_segment_avg	0,327459193719563
28	bwd_length_mean	0,326869934283649
29	bwd_packets	0,319101361263873
30	bwd_header_length	0,316217547740789
31	fwd_iat_max	0,311377950288515
32	total_length_bwd_packet	0,309144184572405
33	fwd_iat_total	0,303618007625008

D. Pelatihan dan Pengujian dengan KNN

Klasifikasi KNN merupakan suatu tahapan dalam machine learning untuk mengelompokkan data berdasarkan data tetangga terdekat atau jarak euclidean. Contoh perhitungan dengan data pada tabel 3 dan tabel 4 dengan KNN dapat dilihat sebagai berikut:

Tabel 3. Contoh sampel data latih perhitungan KNN

src_port	dst_port	protocol	flow_duration	total_fwd_packet	total_bwd_packet	Label
0,978	-0,796	-0,801	-0,547	-0,064	-0,045	NonTOR
0,978	-0,796	-0,801	-0,547	-0,064	-0,045	NonTOR
0,978	-0,796	-0,801	-0,547	-0,064	-0,045	NonTOR
0,558	-0,796	-0,801	-0,547	-0,064	-0,045	TOR
-0,197	0,054	-0,801	-0,264	0,184	0,072	TOR
0,843	-0,796	-0,801	-0,535	-0,062	-0,044	TOR

Tabel 4. Contoh sampel data uji perhitungan KNN

src_port	dst_port	protocol	flow_duration	total_fwd_packet	total_bwd_packet	Label
0,994	-0,7961	-0,80115	-0,536	-0,047	-0,031	NonTOR
0,994	-0,7961	-0,80115	-0,536	-0,063	-0,044	NonTOR
0,492	-0,7961	-0,80115	-0,327	-0,063	-0,044	TOR

Proses klasifikasi KNN dapat dilakukan dengan langkah – langkah sebagai berikut:

- Menentukan nilai k default (jumlah data tetangga terdekat), misalnya k = 3.
- Menghitung nilai euclidean data uji terhadap data latih. Perhitungan dilakukan sebanyak data yang ada pada data latih.

Perhitungan data latih ke-1.

dxy = $\sqrt{}$((0,994-0,978)^2+⋯+ (-0,031-0,045)^2)

= 0,029216

Perhitungan data latih ke – 2.

dxy = $\sqrt{}$((0,994-0,978)^2+⋯+ (-0,031-0,045)^2)

= 0,029216

Perhitungan data latih ke – 3.

dxy = √((0,994-0,978)^2+⋯+ (-0,031-0,045)^2)
= 0,029216

Perhitungan data latih ke – 4.

dxy = √((0,994-0,558)^2+⋯+ (-0,031-0,045)^2)
= 0,436625

Perhitungan data latih ke – 5.

dxy = √((0,994-(-0,197))^2+⋯+ (-0,031-0,072)^2)
= 1,50964

Perhitungan data latih ke – 6.

dxy = √((0,994-0,843)^2+⋯+ (-0,031--0,044)^2)
= 0,15292

Hasil dari perhitungan jarak tersebut diurutkan dari yang terkecil atau secara ascending seperti terlihat pada tabel 5 berikut.

Tabel 5. Hasil perhitungan nilai Euclidian

No Data Latih	*Euclidean*	Label
1	0,029216	TOR
2	0,029216	TOR
3	0,029216	TOR
6	0,15292	NonTOR
4	0,436625	NonTOR
5	1,50964	NonTOR

Klasifikasi data hasil perhitungan menggunakan nilai k yaitu 3, sehingga nomer data latih yang memiliki nilai *euclidean* yang rendah adalah nomer 1, 2 dan 3. Mayoritas data yang dilihat berdasarkan nilai k = 3 adalah label TOR yang dapat menjadi nilai prediksi data uji ke – 1, dengan kata lain data uji ke – 1 termasuk ke dalam kelas TOR. Perhitungan dilakukan secara berulang untuk setiap data uji agar mendapatkan kategori kelas mana yang didapatkan berdasarkan perhitungan jarak tetangga terdekat.

Analisis ancaman serangan *malware* dengan KNN dilakukan dengan membangun model *machine learning* dengan membagi dataset menjadi data latih dan data uji. Data latih berfungsi untuk melatih model sehingga dapat memprediksi variabel dependen dan mengenali pola yang terdapat dalam dataset CICDarknet 2020. Data uji merupakan data yang digunakan untuk menguji tingkat ketepatan atau akurasi dari model *machine learning* yang dibuat.

Proses pembagian data dilakukan dengan membagi dataset CICDarknet 2020 menjadi data latih dan data uji dengan beberapa skenario yang dapat dilihat pada tabel 6 berikut.

Tabel 6. Skenario pembagian data untuk pelatihan dan pengujian

No	Rasio Pembagian Data	
	Data Latih	Data Uji
1	0,8	0,2
2	0,7	0,3
3	0,6	0,4

Hasil pembagian data akan dihitung nilai akurasi nya dengan metode *confussion matrix* untuk mengetahui tingkat akurasi model *machine learning* yang dibangun. Pelatihan dan pengujian dalam penelitian ini dilakukan dua kali dengan menggunakan fitur lengkap dan fitur terpilih hasil seleksi dengan information gain dengan parameter *default* k = 5.

1) Pelatihan dan Pengujian dengan Fitur Lengkap

Pelatihan dan pengujian dengan fitur lengkap dilakukan dengan menggunakan dataset CICDarknet sebelum dilakukan seleksi fitur, hal ini dilakukan untuk melihat perbedaan nilai akurasi berdasarkan jumlah fitur dan rasio pembagian dataset. Hasil perhitungan akurasi dengan fitur lengkap dapat dilihat pada tabel 7 berikut.

Tabel 7. Hasil pelatihan dan pengujian dengan fitur lengkap

No	Rasio		Nilai Akurasi	Waktu Eksekusi
	Data Latih	Data Uji		
1	0,8	0,2	96,38	17.09
2	0,7	0,3	96,22	22.03
3	0,6	0,4	96,07	28.23

Hasil tabel di atas menunjukkan bahwa hasil pelatihan dan pengujian dengan fitur lengkap memiliki nilai akurasi tertinggi adalah 96,38% pada pembagian data dengan rasio 0,8 dan 0,2. Proses pelatihan dan pengujian menghabiskan waktu 17 menit 9 detik. Rasio lainnya menunjukkan bahwa semakin banyak data uji yang diambil maka nilai akurasi akan semakin berkurang dan waktu eksekusi akan semakin lama.

2) Pelatihan dan Pengujian dengan Fitur Terpilih

Pelatihan dan pengujian dengan fitur terpilih dilakukan dengan menggunakan dataset CICDarknet yang telah diseleksi dengan menggunakan information gain, sehingga fitur lebih sedikit namun tetap memperhatikan keterkaitan antara fitur dengan label. Hasil perhitungan akurasi dengan fitur terpilih dapat dilihat pada tabel 8 berikut.

Tabel 8. Hasil pelatihan dan pengujian dengan fitur terpilih

No	Rasio		Nilai Akurasi	Waktu Eksekusi
	Data Latih	Data Uji		
1	0,8	0,2	95,48	06.42
2	0,7	0,3	95,21	09.24
3	0,6	0,4	95,03	10.45

Hasil tabel di atas menunjukkan bahwa nilai akurasi tertinggi adalah 95,48 pada rasio 0.8 dan 0.2 dengan waktu ekseksusi 6 menit 42 detik, selain itu juga dapat dilihat bahwa nilai akurasi dari pelatihan dan pengujian dengan fitur terpilih lebih rendah dari pelatihan dan pengujian dengan fitur lengkap namun memiliki waktu eksekusi yang lebih cepat dibanding fitur lengkap.

3) Pengujian dengan Nilai K

Nilai K dalam algoritma *K-Nearest Neighbour* merupakan hal yang sangat penting yang akan mempengaruhi kinerja KNN, sehingga nilai K dalam KNN perlu diketahui masing – masing akurasinya untuk mendapatkan hasil akurasi yang optimal [14]. Rentang nilai K yang diujikan pada model dalam penelitian ini adalah nilai ganjil dari 1 – 55. Nilai ganjil diperlukan untuk menghindari adanya jumlah label yang sama dari hasil uji. Hasil klasifikasi terbaik ditentukan dengan melihat nilai akurasi terbesar dari setiap nilai K yang diujikan. Perhitungan akurasi dilakukan secara berulang untuk setiap nilai K dengan menggunakan bantuan aplikasi RapidMiner. Hasil perhitungan dapat dilihat pada tabel 9 berikut.

Tabel 9. Hasil pengujian nilai K

Nilai K	Akurasi	Nilai K	Akurasi
1	96,17	29	93,66
3	95,70	31	93,64
5	95,50	33	93,49
7	95,11	35	93,42
9	94,92	37	93,49
11	95,01	39	93,30
13	94,80	41	93,21
15	94,59	43	93,06
17	94,38	45	92,99
19	94,17	47	92,88
21	94,07	49	92,83
23	94,12	51	92,72
25	93,99	53	92,57
27	93,81	55	92,45

Nilai akurasi terbesar adalah pada pengujian dengan nilai K = 1 dengan akurasi 96,17%. Akurasi untuk setiap nilai K memiliki nilai yang beragam, namun terlihat bahwa pada model ini semakin besar nilai K mayoritas nilai akurasi semakin kecil.

E. Evaluasi Model

Evaluasi model dilakukan dengan menggunakan metode *Confussion Matrix* untuk menghitung nilai akurasi, presisi, *recall* dan *f-1 score* dari model yang akan dievaluasi. Akurasi berfungsi untuk menghitung jumlah ukuran proporsi dari klasifikasi yang benar dalam model dengan menggunakan rumus pada persamaan 4.

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN}\ x\ 100\% \quad (4)$$

Presisi berfungsi untuk mengetahui perbandingan jumlah klasifikasi yang benar dengan yang salah dengan menggunakan rumus pada persamaan 5.

$$\text{Presisi} = \frac{TP}{TP+FP}\ x\ 100\% \quad (5)$$

Recall berfungsi untuk menghitung jumlah klasifikasi yang benar dengan melakukan perbandingan terhadap jumlah entri yang terlewat dengan menggunakan rumus pada persamaan 6.

$$Recall = \frac{TP}{TP+ FN}\ x\ 100\% \quad (6)$$

F-1 score berfungsi untuk menghitung rata – rata dari presisi dan recall sehingga dapat diketahui efektivitas dari model yang dibuat dengan menggunakan rumus pada persamaan 7.

$$F\text{-}1\ score = 2\ x\ \frac{Presisi\ x\ Recall}{Presisi+Recall}\ x\ 100\% \quad (7)$$

Kelas positif dan negatif mudah ditemukan ketika label hanya memiliki dua kelas, sehingga untuk kelas yang lebih dari dua nilai TP,TN, FP dan FN harus dicari sesuai masing – masing kelas yang akan dihitung. Perhitungan dengan confussion matrix diterapkan pada dataset hasil pelatihan dengan fitur seleksi serta nilai K yang diambil adalah k = 1. Pengambilan keputusan tersebut didasarkan pada nilai akurasi dan kecepatan eksekusi yang paling tinggi dari setiap pengujian. Hasil data prediksi yang diperoleh dari pelatihan dengan fitur seleksi dan nilai k = 1 dapat dilihat pada tabel 10 berikut.

Tabel 10 Hasil prediksi yang diperolah dari pelatihan dengan fitur seleksi

Hasil Prediksi	*true Non-Tor*	*true Non-VPN*	*true-Tor*	*true-VPN*
pred Non-Tor	18588	96	4	116
pred Non-VPN	71	4265	15	333
pred Tor	1	23	244	9
pred VPN	66	341	8	4116

Akurasi yang didapatkan dari hasil prediksi di atas dapat dilihat sebagai berikut:

$$\text{Akurasi} = \frac{18588 + 4265 + 244 + 4116}{18588 + 4265 + 244 + 4116+96+4+116+71+15+333+1+23+9+66+341+8}\ \text{x}\ 100\%$$

$$= \frac{27213}{28296}\ \text{x}\ 100\% = 96{,}17\ \%$$

Nilai presisi dan *recall* dihitung berdasarkan kelas masing – masing hasil prediksi. Hasil perhitungan dapat dilihat pada tabel 10 berikut.

Tabel 11
Nilai presisi dan recall hasil prediksi setiap kelas

Hasil Prediksi	true Non-Tor	true Non-VPN	true-Tor	true-VPN	Class Precision
pred Non-Tor	18588	96	4	116	98,85%
pred Non-VPN	71	4265	15	333	91,05%
pred Tor	1	23	244	9	88,09%
pred VPN	66	341	8	4116	90,84%
Class Recall	99,26%	90,26%	90,04%	89,99%	

Nilai presisi dan *recall* dari setiap kelas tersebut dihitung rata – ratanya untuk mendapatkan nilai presisi dan recall dari model yang dibuat yang dapat dilihat sebagai berikut.

Presisi = (98,85+91,05+88,09+90,84)/4 = 92,21 %

Recall = (99,26+90,26+90,04+89,99)/4 = 92,39 %

Nilai *f-1 score* dapat dihitung berdasarkan nilai presisi dan recall yang telah diperoleh di atas. Perhitungan nilai f-1 score dapat dilihat sebagai berikut.

F-1 Score = 2 x (92,21 x 92,39)/(92,21+92,39) = 92,30 %

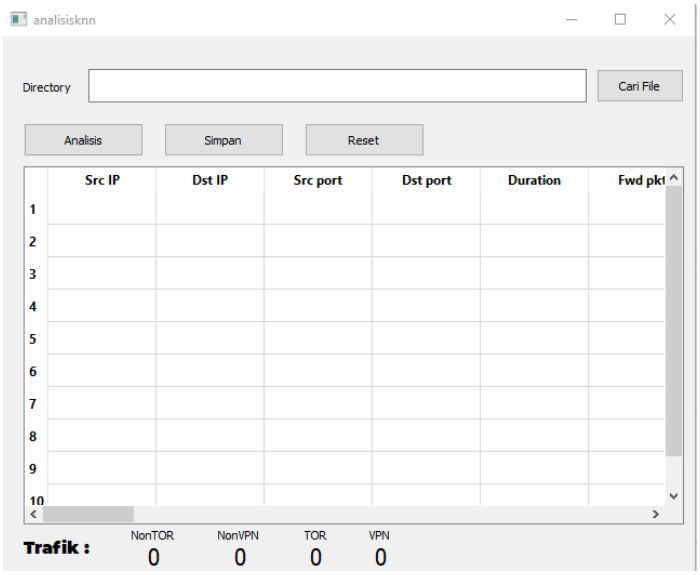
Hasil evaluasi dengan fitur terpilih dan k = 1 mendapatkan nilai akurasi sebesar 96,17% , nilai presisi sebesar 92,21%, nilai *recall* sebesar 92,39% dan nilai f-1 score sebesar 92,30%.

F. Implementasi Sistem

Implementasi sistem dilakukan dengan pembuatan sebuah aplikasi menggunakan bahasa python untuk melakukan analisis trafik darknet secara otomatis.

1) Tampilan Aplikasi Awal

Aplikasi analisis darknet memiliki satu antarmuka yang didalamnya terdapat empat tombol yang berfungsi untuk mencari file yang akan dianalisis, melakukan analisis, menyimpan hasil analisis dan melakukan reset aplikasi yang terlihat seperti pada gambar 5.

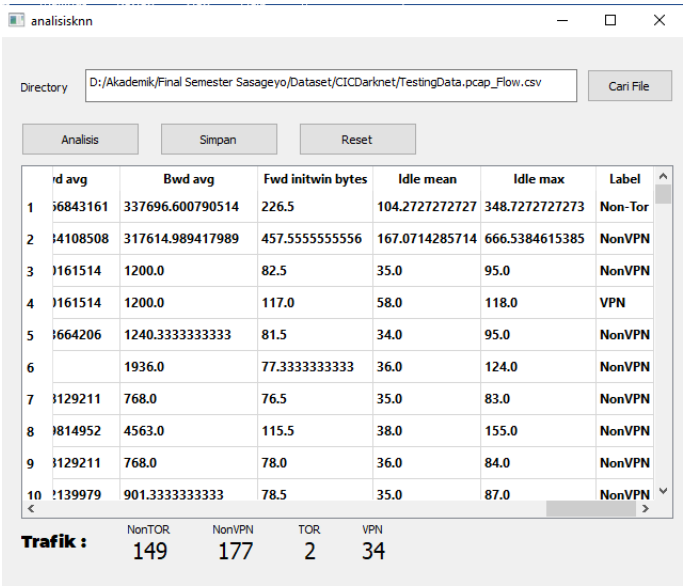


Gambar 5. Tampilan Aplikasi Awal

Data tabel dalam aplikasi secara default akan kosong dikarenakan belum dilakukannya proses analisis. Tabel tersebut memuat semua nama kolom berdasarkan data yang ada pada file analisis.

2) Tampilan Aplikasi Hasil

Aplikasi analisis darknet yang telah dilakukan analisis akan menampilkan tabel prediksi dari setiap paket data di dalam file yang dianalisis. Hasil prediksi menjadi acuan paket data mana saja yang perlu dianalisis lebih lanjut karena memuat trafik darknet yang dapat menjadi ancaman serangan malware. Aplikasi hasil dari analisis dapat dilihat pada gambar 6 berikut.



Gambar 6. Tampilan Aplikasi Hasil

Hasil tabel akan muncul setelah proses analisis selesai, semua kolom akan terisi dengan data masukan beserta prediksi yang dihasilkan menggunakan algoritma KNN. Aplikasi ini memberikan informasi jumlah trafik data yang diprediksi sebagai darknet ataupun sebagai trafik biasa.

IV. KESIMPULAN

Penelitian ini dilakukan dengan melakukan tahapan pemodelan *machine learning* untuk menganalisa ancaman dini terhadap serangan *malware* dengan melakukan klasifikasi dan prediksi trafik *darknet* dengan menggunakan algoritma KNN. Proses seleksi fitur dilakukan untuk mengefesienkan dataset yang memiliki banyak fitur dengan memperhatikan nilai *information gain* dari setiap fitur. Fitur yang telah diseleksi berjumlah 33 fitur dengan nilai *information gain* diatas 0,3. Proses pengujian dilakukan dengan membagi dataset menjadi data latih dan data uji dengan rasio terbaik adalah 0,8 data latih dan 0,2 data uji dan memiliki akurasi 95,48% serta waktu eksekusi 6 menit 42 detik. Hasil evaluasi dengan fitur terpilih untuk mendapatkan nilai k terbaik dilakukan secara berulang untuk nilai k antara 1- 55, sehingga didapatkan k terbaik = 1 dengan nilai akurasi sebesar 96,17% , nilai presisi sebesar 92,21%, nilai *recall* sebesar 92,39% dan nilai f-1 score sebesar 92,30%. Berdasarkan nilai tersebut dapat disimpulkan bahwa dataset CICDarknet dengan fitur terpilih dapat digunakan untuk melakukan klasifikasi dan prediksi trafik *darknet* pada proses *network forensic.*

DAFTAR PUSTAKA

[1] F. A. Basyarahil, H. M. Astuti, and C. Hidayanto, "… Keamanan Informasi Menggunakan Indeks Keamanan Informasi Direktorat Pengembangan Teknologi dan Sistem Informasi (DPTSI) ITS Surabaya," *J. Tek. ITS*, vol. 6, no. 1, 2017.

[2] A. Ramadhani, "Keamanan Informasi," *Nusant. - J. Inf. Libr. Stud.*, vol. 1, no. 1, p. 39, 2018, doi: 10.30999/n-jils.v1i1.249.

[3] A. Ginanjar, N. Widiyasono, and R. Gunawan, "Web Phising Attack Analysis on E-

Commerce Service Using Network Forensic Process Method," *J. Terap. Teknol. Inf.*, vol. 2, no. 2, pp. 59–69, 2019, doi: 10.21460/jutei.2018.22.111.

[4] T. A. Cahyanto, V. Wahanggara, and D. Ramadana, "Analisis dan Deteksi Malware Menggunakan Metode Malware Analisis Dinamis dan Malware Analisis Statis," *J. Sist. dan Teknol. Inf. Indones.*, vol. 2, pp. 19–30, 2017.

[5] V. R. Kebande and I. Ray, "A generic digital forensic investigation framework for Internet of Things (IoT)," *Proc. - 2016 IEEE 4th Int. Conf. Futur. Internet Things Cloud, FiCloud 2016*, pp. 356–362, 2016, doi: 10.1109/FiCloud.2016.57.

[6] A. Lakoro, L. W. Badu, and N. Achir, "PERJUDIAN TOGEL ONLINE ' Weak Polices In Handling Criminal Actions Online Togel Gaming ."

[7] T. P. Setia, A. P. Aldya, and N. Widiyasono, "Reverse Engineering untuk Analisis Malware Remote Access Trojan," *J. Edukasi dan Penelit. Inform.*, vol. 5, no. 1, p. 40, 2019, doi: 10.26418/jp.v5i1.28214.

[8] T. Ban, L. Zhu, J. Shimamura, S. Pang, D. Inoue, and K. Nakao, "Behavior analysis of long-term cyber attacks in the darknet," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7667 LNCS, no. PART 5, pp. 620–628, 2012, doi: 10.1007/978-3-642-34500-5_73.

[9] S. Kumar, H. Vranken, J. Van DIjk, and T. Hamalainen, "Deep in the Dark: A Novel Threat Detection System using Darknet Traffic," *Proc. - 2019 IEEE Int. Conf. Big Data, Big Data 2019*, pp. 4273–4279, 2019, doi: 10.1109/BigData47090.2019.9006374.

[10] F. Rolansa, Y. Yunita, and S. Suheri, "Sistem prediksi dan evaluasi prestasi akademik mahasiswa di Program Studi Teknik Informatika menggunakan data mining," *J. Pendidik. Inform. dan Sains*, vol. 9, no. 1, p. 75, 2020, doi: 10.31571/saintek.v9i1.1696.

[11] E. Etriyanti, D. Syamsuar, and N. Kunang, "Implementasi Data Mining Menggunakan Algoritme Naive Bayes Classifier dan C4.5 untuk Memprediksi Kelulusan Mahasiswa," *Telematika*, vol. 13, no. 1, pp. 56–67, 2020, doi: 10.35671/telematika.v13i1.881.

[12] D. A. Nasution, H. H. Khotimah, and N. Chamidah, "Perbandingan Normalisasi Data untuk Klasifikasi Wine Menggunakan Algoritma K-NN," *Comput. Eng. Sci. Syst. J.*, vol. 4, no. 1, p. 78, 2019, doi: 10.24114/cess.v4i1.11458.

[13] Kurniabudi, A. Harris, and A. Rahim, "Seleksi Fitur dengan Information Gain untuk Meningkatkan Deteksi Serangan DDoS Menggunakan Random Forest," vol. 19, no. 1, pp. 56–66, 2020.

[14] M. A. Banjarsari, H. I. Budiman, and A. Farmadi, "Penerapan K-Optimal Pada Algoritma Knn Untuk Prediksi Kelulusan Tepat Waktu Mahasiswa Program Studi Ilmu Komputer Fmipa Unlam Berdasarkan Ip Sampai Dengan Semester 4," *Klik - Kumpul. J. Ilmu Komput.*, vol. 2, no. 2, pp. 159–173, 2015.