INVESTIGASI LOG JARINGAN UNTUK DETEKSI SERANGAN *DISTRIBUTED DENIAL OF SERVICE* (DDOS) DENGAN MENGGUNAKAN METODE *GENERAL REGRESSION NEURAL NETWORK*

Muhammad Hilmi Hafid

Abstract—**One type of attack in cyberspace with a large enough intensity is the Distributed Denial of Service (DDoS) attack. To overcome this, an effective and accurate intrusion detection system needed to detect attacks on network intrusion data. Therefore, this study aims to implement a new approach in detecting attacks on network intrusion data with a good rate of accuracy. The proposed method is the General Regression Neural Network which is assisted by Random Forest in selecting features so as to improve detection accuracy and speed up computing time. The training data used is CICIDS2017 from the Canadian Institute for Cybersecurity, while the test data used are network logs obtained from a simulation of DDoS attacks on a web server. The first experiment using 69 features, obtained an accuracy rate of 66.41% with training time for 1 hour 45 minutes 6 seconds. As for the second experiment using selected features which is 20 features, obtained an accuracy rate of 97.21% with training time for 42 minutes 27 seconds. From the results of these experiments, it can be concluded that the General Regression Neural Network has a fairly good detection and classification ability against DDoS attacks on network intrusion data.**

Index Terms—**network forensics, intrusion detection system, machine learning, distributed denial of service, general regression neural network, random forest.**

Abstrak—**Salah satu jenis serangan di dunia maya dengan intensitas yang cukup besar yaitu serangan *Distributed Denial of Service* (DDoS). Dibutuhkan sistem deteksi intrusi yang efektif dan akurat dalam mendeteksi serangan pada data intrusi jaringan untuk mengatasi hal tersebut. Oleh sebab itu, penelitian ini bertujuan untuk mengimplementasikan pendekatan baru dalam mendeteksi serangan pada data intrusi jaringan dengan tingkat akurasi yang baik. Metode yang diusulkan yaitu *General Regression Neural Network* yang dibantu dengan *Random Forest* dalam menyeleksi fitur sehingga mampu meningkatkan akurasi deteksi dan mempercepat waktu komputasi. Data latih yang digunakan yaitu CICIDS2017 dari Canadian Institute for Cybersecurity, sedangkan data uji yang digunakan yaitu log jaringan yang didapatkan dari simulasi serangan DDoS pada server web. Percobaan pertama menggunakan 69 fitur, diperoleh tingkat akurasi sebesar 66,41% dengan waktu pelatihan selama 1 jam 45 menit 6 detik. Adapun percobaan kedua menggunakan fitur terpilih yaitu sebanyak 20 fitur, diperoleh tingkat akurasi sebesar 97,21% dengan waktu pelatihan selama 42 menit 27 detik. Dari hasil percobaan tersebut, dapat disimpulkan bahwa *General Regression Neural Network* memiliki kemampuan deteksi dan klasifikasi yang cukup baik terhadap serangan DDoS pada data intrusi jaringan.**

Kata Kunci—**forensik jaringan, sistem deteksi intrusi, *machine learning, distributed denial of service*, *general regression neural network*, *random forest*..**

I. PENDAHULUAN

Seiring dengan pesatnya perkembangan teknologi jaringan komputer dan internet, banyak institusi saat ini yang menyalurkan layanannya dengan memanfaatkan fasilitas intemet. Selain untuk mempermudah kinerja, institusi terkait juga dapat memperoleh keuntungan yang lebih karena di era saat ini internet sudah menjadi kebutuhan utama bagi sebagian orang. Internet di mana dan kapan pun dapat digunakan, seperti mengakses informasi, menghubungi kerabat, bahkan jual beli barang.

Dari hasil survei Asosiasi Penyelenggara Jasa Intemet Indonesia (APJII) di tahun 2019, menunjukkan bahwa jumlah pengguna intemet di Indonesia pada tahun 2017 yaitu sebanyak 143,26 juta dari 262 juta total populasi penduduk Indonesia atau sebesar 54,68%. Jauh berbeda dengan tahun 2018, terhitung sebanyak 171,17 juta pengguna internet dari 264,16 juta total populasi penduduk Indonesia atau sebesar 64.8%. Dapat disimpulkan bahwa penetrasi pengguna internet di Indonesia dari tahun ke tahun terus meningkat.

Dengan berkembang pesatnya pengguna internet, maka tidak heran kasus kejahatan di dunia maya juga terus meningkat. Ada beberapa kasus serangan pada web yang sering terjadi, dan yang mendominasi di antaranya adalah serangan *SQL Injection*, *Cross Site Scripting* (XSS), *Denial of Service* (DoS) dan lain sebagainya [1].

Salah satu jenis serangan dengan intensitas yang cukup besar yaitu serangan *Denial of Service*. Serangan *Denial of Service* merupakan percobaan yang dilakukan oleh peretas untuk melumpuhkan sistem target dengan cara menghabiskan jaringan atau sumber daya dari sistem tersebut. Jika serangan ini dilakukan dengan lebih dari satu mesin, maka disebut dengan serangan *Distributed Denial of Service* (DDoS) [2]. Serangan DDoS adalah salah satu ancaman paling serius untuk keamanan jaringan, dan jumlah korban serangan DDoS terus meningkat setiap harinya [3].

Salah satu serangan DDoS yang terbesar terjadi pada tanggal 5 Maret 2018 menyerang penyedia layanan internet yang berbasis di USA. Serangan ini berhasil dideteksi oleh peneliti dari Arbor Networks, penyedia layanan pencegahan serangan DDoS dengan analisis perkiraan data yang dikirimkan mencapai 1.7 Tbps [4].

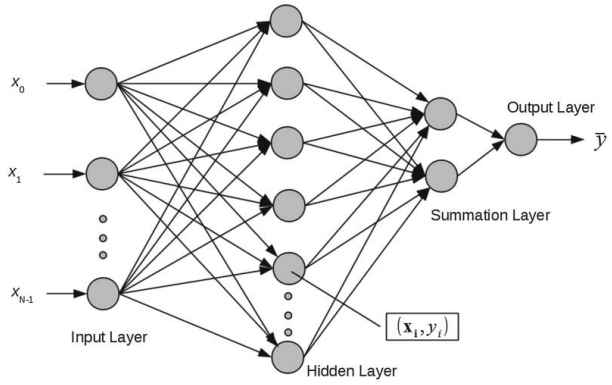
Dari latar belakang tersebut, maka penelitian ini diharapkan mampu mengurangi kejahatan di dunia maya dengan menghasilkan persentase akurasi deteksi yang lebih baik dan mampu meningkatkan performa sistem deteksi intrusi jaringan dengan mengimplementasikan *General Regression Neural Network.*

II. LANDASAN TEORI

2.1 *General Regression Neural Network* (GRNN)

General Regression Neural Network merupakan variasi dari *Radial Basis Neural Network* yang diusulkan pertama kali oleh D.F. Specht in 1991. *General Regression Neural Network* dapat digunakan pada kasus regresi, prediksi, maupun klasifikasi serta dapat menjadi solusi yang baik untuk sistem dinamis [5].

General Regression Neural Network memiliki empat lapisan unit pemrosesan. Lapisan-lapisan tersebut yaitu lapisan masukan (*input layer*), lapisan tersembunyi (*hidden layer*), lapisan penjumlahan (*summation layer*), dan yang terakhir lapisan keluaran (*output layer*). Arsitektur *General Regression Neural Network* [6] dapat dilihat pada Gambar 1.



Gambar 1 Arsitektur GRNN

Setiap lapisan unit pemrosesan ditandai dengan suatu fungsi yang spesifik [7].

1. Lapisan masukan, menyimpan neuron masukan untuk setiap variabel prediktor tanpa adanya pemrosesan data yang dilakukan. Tidak ada pemrosesan data yang dilakukan pada neuron-neuron tersebut. Neuron masukan kemudian mengirimkan data ke lapisan kedua dari unit pemrosesan yang disebut lapisan tersembunyi.
2. Lapisan tersembunyi, menyimpan nilai-nilai dari variabel-variabel prediktor bersama dengan nilai target. Dalam hal ini, neuron pada lapisan tersembunyi sama dengan neuron pada lapisan masukan. Neuron pada lapisan tersembunyi menghitung matriks jarak dari nilai target sebagai titik pusat neuron, kemudian menerapkan fungsi kernel radial basis menggunakan nilai sigma (σ).
3. Lapisan penjumlahan, hanya memiliki dua neuron yaitu neuron penjumlahan penyebut dan neuron penjumlahan pembilang. Neuron penjumlahan penyebut menambah bobot yang berasal dari masing-masing neuron tersembunyi. Sedangkan neuron penjumlahan pembilang menambah nilai bobot dikalikan nilai target aktual untuk setiap neuron tersembunyi.
4. Lapisan keluaran, membagi nilai akumulasi dalam unit penjumlahan pembilang dengan nilai di unit penjumlahan penyebut yang kemudian hasilnya ditetapkan sebagai nilai terget prediksi.

Adapun persamaan matematis dari *General Regression Neural Network* [5] dapat dilihat pada Persamaan 1.

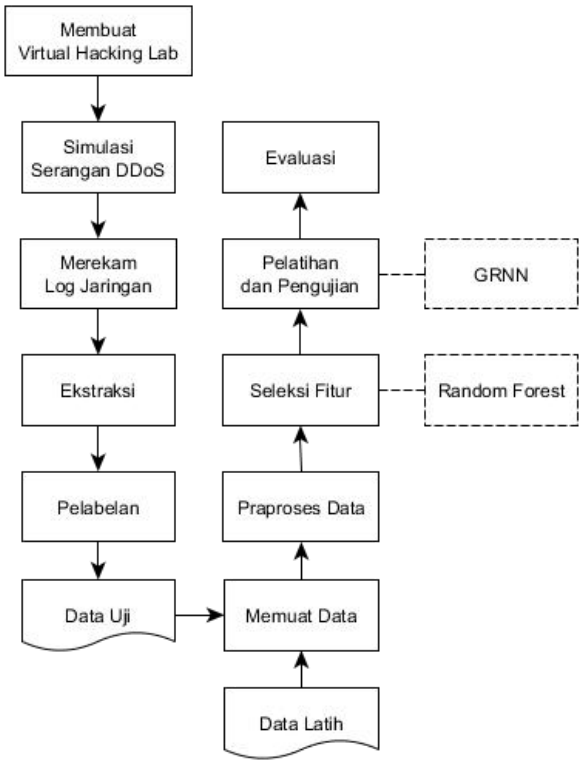
$$\hat{Y}(X) = \frac{\sum_{i=1}^{n} Y_i \exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]}{\sum_{i=1}^{n} \exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]} \quad (1)$$

Dimana:

1. X adalah sampel masukan dan X_i adalah sampel pengujian.
2. Keluaran dari sampel masukan *i* adalah Y_i.
3. $(X-X_i)^T(X-X_i)$ adalah jarak Euclidean dari X dan X_i.
4. $\exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]$ merupakan fungsi aktivasi *radial basis function.* Sederhananya, fungsi aktivasi ini secara teori merupakan bobot untuk sampel masukan.
5. Nilai dari $(X-X_i)^T(X-X_i)$ menandakan seberapa banyak sampel pelatihan dapat berkontribusi pada keluaran dari sampel uji tertentu.
6. Jika $(X-X_i)^T(X-X_i)$ memiliki yang nilai kecil, berarti akan memberikan kontribusi nilai lebih untuk keluaran. Tetapi jika memiliki nilai yang besar, berarti akan memberikan kontribusi lebih sedikit ke keluaran.
7. Adapun $\exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]$ menentukan seberapa banyak berat sampel pelatihan akan berkontribusi.
8. Jika $(X-X_i)^T(X-X_i)$ menghasilkan nilai kecil, maka $\exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]$ mengembalikan nilai yang relatif besar dan begitupun sebaliknya.
9. Jika $(X-X_i)^T(X-X_i)$ menghasilkan nilai 0, maka $\exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]$ mengembalikan nilai 1 yang berarti data uji sama dengan sampel pelatihan dan keluaran dari data uji juga merupakan keluaran dari sampel pelatihan.

III. METODOLOGI PENELITIAN

Penelitian ini melalui beberapa tahapan seperti yang dapat dilihat pada Gambar 2.



Gambar 2 Prosedur Penelitian

Tahapan-tahapan tersebut yaitu sebagai berikut: 1) Membuat *virtual hacking lab* sederhana dengan menggunakan Oracle VM VirtualBox. 2) Melakukan simulasi serangan DDoS ke server web dengan menggunakan aplikasi *Low Orbit Ion Cannon* (LOIC) dan Metasploit. 3) Disaat serangan berlangsung, log jaringan dipantau dan direkam menggunakan Wireshark. 4) Data dengan format PCAP yang didapatkan sebelumnya kemudian diekstrak menggunakan aplikasi CICFlowMeter 4.0. Dari hasil ekstraksi diperoleh data dengan format CSV yang akan digunakan sebagai data uji. 5) Dilakukan pelabelan untuk memberi kelas pada data uji berdasarkan skema waktu serangan yang dilakukan. 6) Tahap selanjutnya yaitu memuat data latih dari set data CICIDS2017 dan data uji dari simulasi DDoS yang dilakukan sebelumnya. 7) Setelah itu dilakukan praproses data yang meliputi penghapusan sebagian data yang tidak perlu dan normalisasi keseluruhan data. 8) Sebelum dilakukan pelatihan dan pengujian, perlu dilakukan pemilihan fitur untuk meningkatkan akurasi dan mengurangi waktu komputasi. Algoritma yang digunakan untuk pemilihan fitur yaitu *Random Forest.* 9) Kemudian dilakukan analisis pelatihan dan pengujian menggunakan *General Regression Neural Network.* 10) Setelah itu dilakukan evaluasi model untuk mengetahui persentase *accuracy*, *precision*, *recall*, dan *f1-score* melalui c*onfusion matrix.*

3.1 Membuat *Virtual Hacking Lab*

Virtual hacking lab dibuat dengan menggunakan Oracle VM VirtualBox sebagai mesin virtual, sistem operasi Windows dan Kali Linux sebagai penyerang, dan Metasploitable yang akan bertindak sebagai server web. Semua perangkat tersebut dikonfigurasikan sehingga dapat terhubung satu sama lain.

3.2 Simulasi Serangan DDoS

Salah satu aplikasi yang digunakan untuk simulasi serangan DDoS yaitu *Low Orbit Ion Cannon* (LOIC). Bisa dibilang jika LOIC adalah aplikasi yang paling populer digunakan untuk melakukan serangan DoS. Aplikasi berbasis Windows ini efektif untuk mengirimkan banyak jumlah paket ICMP, TCP atau UDP ke target. LOIC telah teruji digunakan oleh 4chan dalam serangan *project chanology* terhadap web Church of Scientology di tahun 2009. Selain itu LOIC juga digunakan oleh Anonymous dalam *operation payback* melawan PayPal, Visa, and MasterCard karena memotong aliran dana sumbangan ke Wikileaks.

Selain LOIC, Metasploit juga digunakan untuk melakukan simulasi serangan DDoS. Metasploit adalah proyek keamanan komputer yang menyediakan informasi tentang kerentanan keamanan dan bantuan dalam pengujian penetrasi dan pengembangan sistem deteksi intrusi.

Kedua aplikasi tersebut digunakan dalam waktu bersamaan. Adapun simulasi serangan dilakukan dengan memperhatikan skema waktu serangan. Simulasi serangan dilakukan dengan selang waktu beberapa menit untuk penyerangan dan beberapa menit pula untuk menormalkan lalu lintas jaringan. Hal ini dilakukan berulang kali selama kurang lebih 20 menit.

3.3 Merekam Log Jaringan

Wireshark merupakan *network protocol analyzer* alias penganalisa protokol jaringan yang memiliki fitur lengkap. Aplikasi ini dapat menangkap semua paket yang lewat serta menyeleksi dan menampilkan data tersebut sedetail mungkin.

Wireshark merekam semua paket yang melewati *interface* yang dipilih. *Interface* adalah perangkat penghubung antar jaringan, bisa melalui *wifi* atau *ethernet*. Hasil rekaman tadi dapat dianalisa dengan memfilter protokol apa yang ingin ditampilkan seperti TCP, HTTP, UDP dan sebagainya serta dapat mencatat *cookie*, *post* dan *request*. Log jaringan yang direkam dapat disimpan dalam beberapa format, salah satunya PCAP.

3.4 Ekstraksi

Untuk mengekstrak fitur lalu lintas jaringan, aplikasi yang digunakan yaitu CICFlowMeter yang merupakan ekstraktor fitur dan dapat mengekstrak lebih dari 80 fitur dari data dengan format PCAP. CICFlowMeter merupakan aplikasi sumber terbuka yang dikembangkan oleh Canadian Institute for Cybersecurity. Aplikasi ini telah banyak digunakan di banyak set data dalam bidang *cyber security.*

CICFlowMeter menghasilkan *Bidirectional Flows* (Biflow), di mana paket pertama menentukan arah maju (sumber ke tujuan) dan mundur (tujuan ke sumber), maka 84 fitur statistik seperti *Duration*, *Number of packets*, *Number of bytes*, *Length of packets*, dan lainnya juga dihitung secara terpisah dalam arah maju dan mundur. Output dari aplikasi ini adalah data dengan format CSV dengan enam kolom berlabel untuk setiap aliran, yaitu *FlowID*, *SourceIP*, *DestinationIP*, *SourcePort*, *DestinationPort*, dan *Protocol* dengan lebih dari 80 fitur lalu lintas jaringan [8].

3.5 Pelabelan

Proses ekstraksi menghasilkan data dengan format CSV yang berisi fitur-fitur yang sama seperti data latih. Tetapi dari hasil ekstraksi yang dilakukan, fitur *Label* tidak dapat diklasifikasikan secara otomatis, mana yang termasuk kelas normal ataupun kelas serangan. Meskipun demikian, pelabelan dilakukan secara manual dengan memperhatikan skema waktu simulasi serangan yang dilakukan sebelumnya.

3.6 Memuat Data Latih dan Data Uji

a. Data Latih

Data yang digunakan yaitu set data CICIDS2017 yang dikeluarkan oleh Canadian Institute for Cybersecurity. Set data ini sejak awal mulai menarik para peneliti untuk melakukan analisis serta pengembangan model dan algoritma baru [9]. Set data CICIDS2017 berisi kelas normal dan serangan terbaru pada jaringan dalam format PCAP. Selain itu tersedia juga data dengan format CSV yang sudah diekstrak menggunakan CICFlowMeter dengan tujuan pembelajaran *machine learning* atau *deep learning* yang dapat dengan bebas digunakan oleh peneliti.

Set data ini menyediakan data lengkap berisi berbagai jenis serangan pada jaringan dan juga data terpisah setiap serangan. Data latih yang digunakan pada penelitian ini yaitu data yang khusus berisi kelas normal dan kelas serangan DDoS saja. Deskripsi singkat dari data yang digunakan dapat dilihat pada Tabel 1. Adapun fitur-fitur yang ada pada data ini yaitu sebanyak 84 fitur seperti yang dapat dilihat pada Tabel 2.

Tabel 1 Deskripsi Singkat Set Data Latih yang Digunakan

Nama Data	Friday-WorkingHours-Afternoon-DDos.pcap_ISCX
Tahun Rilis	2017
Jenis Serangan	DDoS LOIT
Jumlah Data	225.745
Jumlah Fitur	84

Tabel 2 Fitur-Fitur Set Data

No	Nama Fitur	No	Nama Fitur
1.	Flow ID	43.	Fwd Pkts/s
2.	Src IP	44.	Bwd Pkts/s
3.	Src Port	45.	Pkt Len Min
4.	Dst IP	46.	Pkt Len Max
5.	Dst Port	47.	Pkt Len Mean
6.	Protocol	48.	Pkt Len Std
7.	Timestamp	49.	Pkt Len Var
8.	Flow Duration	50.	FIN Flag Cnt
9.	Tot Fwd Pkts	51.	SYN Flag Cnt
10.	Tot Bwd Pkts	52.	RST Flag Cnt
11.	TotLen Fwd Pkts	53.	PSH Flag Cnt
12.	TotLen Bwd Pkts	54.	ACK Flag Cnt
13.	Fwd Pkt Len Max	55.	URG Flag Cnt
14.	Fwd Pkt Len Min	56.	CWE Flag Count
15,	Fwd Pkt Len Mean	57.	ECE Flag Cnt
16.	Fwd Pkt Len Std	58.	Down/Up Ratio
17.	Bwd Pkt Len Max	59.	Pkt Size Avg
18.	Bwd Pkt Len Min	60.	Fwd Seg Size Avg
19.	Bwd Pkt Len Mean	61.	Bwd Seg Size Avg
20.	Bwd Pkt Len Std	62.	Fwd Byts/b Avg
21.	Flow Byts/s	63.	Fwd Pkts/b Avg
22.	Flow Pkts/s	64.	Fwd Blk Rate Avg
23.	Flow IAT Mean	65.	Bwd Byts/b Avg
24.	Flow IAT Std	66.	Bwd Pkts/b Avg
25.	Flow IAT Max	67.	Bwd Blk Rate Avg
26.	Flow IAT Min	68.	Subflow Fwd Pkts
27.	Fwd IAT Tot	69.	Subflow Fwd Byts
28.	Fwd IAT Mean	70.	Subflow Bwd Pkts
29.	Fwd IAT Std	71.	Subflow Bwd Byts
30.	Fwd IAT Max	72.	Init Fwd Win Byts
31.	Fwd IAT Min	73.	Init Bwd Win Byts
32.	Bwd IAT Tot	74.	Fwd Act Data Pkts
33.	Bwd IAT Mean	75.	Fwd Seg Size Min
34.	Bwd IAT Std	76.	Active Mean
35.	Bwd IAT Max	77.	Active Std
36.	Bwd IAT Min	78.	Active Max
37.	Fwd PSH Flags	79.	Active Min
38.	Bwd PSH Flags	80.	Idle Mean
39.	Fwd URG Flags	81.	Idle Std
40.	Bwd URG Flags	82.	Idle Max
41.	Fwd Header Len	83.	Idle Min
42.	Bwd Header Len	84.	Label

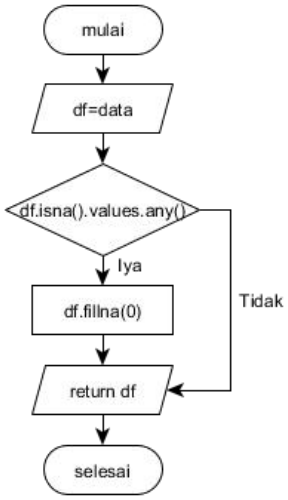
b. Data Uji

Data uji yang digunakan adalah hasil dari simulasi serangan DDoS terhadap server web. Bersamaan dengan dilakukannya simulasi serangan, lalu lintas jaringan dipantau dan direkam menggunakan Wireshark. Data yang didapatkan awalnya merupakan data log jaringan dengan format PCAP yang kemudian dilakukan ekstraksi dengan menggunakan CICFlowMeter 4.0 untuk mendapatkan format data yang sama dengan data latih.

3.7 Praproses Data

a. Pembersihan Data

Tahap praproses yang pertama dilakukan yaitu pembersihan data. Pembersihan data perlu dilakukan karena sering didapati data yang hilang atau rusak. Penting untuk memahami sumber data yang hilang atau rusak tersebut. Bisa jadi kesalahan pada saat transfer data, kesalahan yang disebabkan oleh sistem ataupun disebabkan oleh permasalahan lainnya. Jika hal tersebut ada pada data yang digunakan, maka akan menghambat proses analisis yang dilakukan. Oleh sebab itu data tersebut perlu dihilangkan dengan cara mengganti data tersebut dengan nilai konstan atau dengan nilai tengah atau rata-rata dari data yang sekolom dengan data tersebut. *Flowchart* dari pembersihan data ditunjukkan pada Gambar 3.



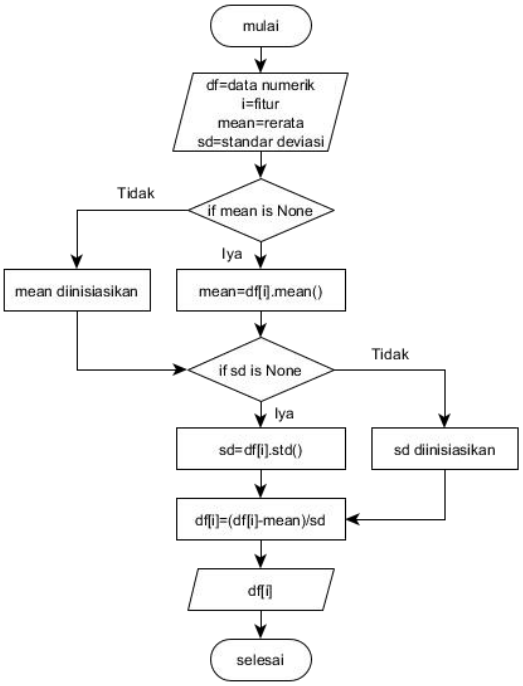
Gambar 3 *Flowchart* Membersihkan Data

b. Normalisasi Atribut Numerik

CICIDS2017 sebagai set data yang digunakan, memiliki tiga jenis tipe data yaitu *integer*, *float*, dan *object.* Normalisasi atribut numerik dilakukan dengan melakukan standarisasi data pada semua atribut dengan tipe data *integer* dan *float.* Atribut numerik yang ada memiliki nilai dengan rentang berbeda-beda. Ada yang jumlahnya besar dan ada pula yang kecil. Rata-rata dan deviasi standar dihitung pada setiap fitur dan kemudian fitur dinormalisasikan berdasarkan pada persamaan 2:

$$\text{normalisasi}(x_i) = \frac{x_i - mean(x)}{stdev(x)} \qquad (2)$$

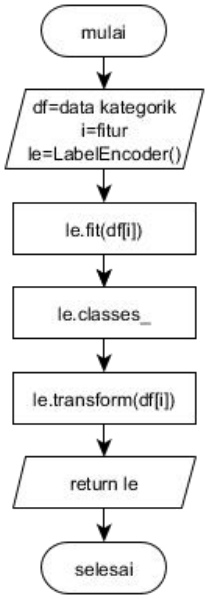
Normalisasi atribut numerik ini dilakukan agar nilai tersebut menjadi lebih kecil dan seragam tanpa mengubah esensi dari nilai tersebut. Dengan demikian data dapat dianalisis dengan baik oleh model jaringan saraf tiruan yang digunakan. *Flowchart* normalisasi atribut numerik ditunjukkan pada Gambar 4.



Gambar 4 *Flowchart* Normalisasi Atribut Numerik

c. Normalisasi Atribut Kategorik

Atribut kategorik pada set data CICIDS2017 terdapat pada kolom Label. Pada kolom tersebut terdapat dua kategori yang ada yaitu kelas normal dan kelas serangan. Seperti yang diketahui, jaringan saraf tiruan tidak dapat mengenali data yang berbentuk teks yang dalam hal ini disebut atribut kategorik. Jadi sebelum jaringan saraf tiruan bekerja, atribut kategorik harus dinormalisasi menjadi atribut numerik terlebih dahulu. Dengan dilakukannya normalisasi, yang awalnya merupakan kelas normal akan berubah menjadi angka 0, sedangkan yang awalnya merupakan kelas serangan akan berubah menjadi angka 1. *Flowchart* normalisasi atribut kategorik ditunjukkan Gambar 5.

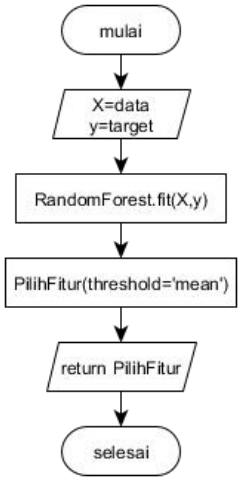


Gambar 5 *Flowchart* Normalisasi Atribut Kategorik

3.8 Seleksi Fitur

Dalam pemilihan fitur-fitur yang ada pada data latih dan data uji, algoritma yang digunakan yaitu *Random Forest*. Algoritma ini sering digunakan untuk pemilihan fitur karena strategi berbasis pohon yang digunakan secara alami memberi penilaian dengan seberapa baik mereka meningkatkan kemurnian simpul. Hal ini mengurangi rata-rata ketidakmurnian di semua pohon (disebut *Gini Impurity*). Simpul dengan penurunan ketidakmurnian terbesar terjadi di awal pohon, sementara catatan dengan penurunan ketidakmurnian paling sedikit terjadi di ujung pohon. Dengan demikian, dengan memangkas pohon di bawah simpul tertentu, kita dapat membuat bagian dari fitur yang paling penting.

Setelah menilai seberapa penting fitur yang ada pada set data dengan melakukan pelatihan pada setiap pohon, maka dipilih beberapa fitur yang dianggap penting. Fitur yang dipilih berdasarkan kaidah standar pemilihan fitur yaitu berdasarkan ambang batas rata-rata nilai dari setiap fitur. *Flowchart* seleksi fitur dengan menggunakan *Random Forest* ditunjukkan pada Gambar 6.



Gambar 6 *Flowchart* Seleksi Fitur dengan *Random Forest*

3.9 Pelatihan dan Pengujian dengan GRNN

Pelatihan dilakukan dengan mengimplementasikan teknik *Cross Validation* untuk mendapatkan parameter sigma (σ) terbaik. Jumlah iterasi yang diterapkan yaitu sebanyak 15 kali dengan penambahan sigma sebesar 0.1 di setiap iterasi. Setelah mendapatkan sigma terbaik, maka selanjutnya dilakukan pengujian. Percobaan dilakukan sebanyak dua kali, yaitu pelatihan dan pengujian dengan fitur lengkap serta pelatihan dan pengujian dengan fitur terpilih.

3.10 Evaluasi

Evaluasi dilakukan dengan menggunakan *Confusion Matrix* yang merupakan metode yang biasanya digunakan dalam evaluasi model pada kasus klasifikasi untuk menghitung tingkat *accuracy*, *precision, recall*, dan *f1-Score.*

Accuracy mengukur proporsi jumlah total klasifikasi yang benar. Rumus dari *accuracy* ditunjukkan pada persamaan 3.8.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \qquad (3.8)$$

Precision mengukur jumlah klasifikasi yang benar dibandingkan dengan jumlah klasifikasi yang salah. Rumus *precision* ditunjukkan pada persamaan 3.9.

$$Precision = \frac{TP}{TP + FP} \times 100\% \qquad (3.9)$$

Recall mengukur jumlah klasifikasi yang benar dibandingkan dengan jumlah entri yang terlewat. Rumus dari *recall* ditunjukkan pada persamaan 3.10.

$$Recall = \frac{TP}{TP + FN} \times 100\% \qquad (3.10)$$

F1-score mengukur rata-rata *precision* dan *recall*, yang berfungsi sebagai pengukuran efektivitas. Rumus dari *F1-score* ditunjukkan pada persamaan 3.11.

$$F1\text{-}score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \times 100\% \qquad (3.11)$$

Untuk lebih mudah memahami, berikut adalah tabel kebenaran antara nilai sebenarnya dan nilai prediksi seperti yang ditunjukkan pada Tabel 3.

Tabel 3 Tabel Kebenaran Antara Nilai Sebenarnya dan Nilai Prediksi

Nilai Sebenarnya/ Nilai Prediksi	**Normal**	**Serangan**
Normal	TP	FP
Serangan	FN	TN

Dimana:
- *True Positive* (TP) adalah kelas serangan yang diklasifikasikan dengan benar sebagai serangan.
- *False Positive* (FP) adalah kelas normal yang salah diklasifikasikan sebagai serangan.
- *True Negative* (TN) adalah kelas normal yang diklasifikasikan dengan benar sebagai normal.
- *False Negative* (FN) adalah kelas serangan yang salah diklasifikasikan sebagai normal.

IV. HASIL DAN PEMBAHASAN

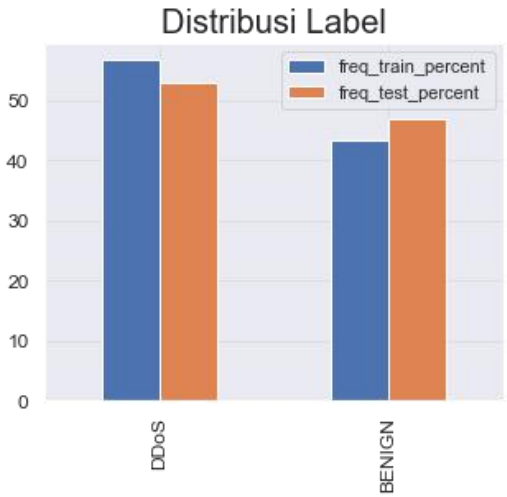
Simulasi serangan DDoS dilakukan untuk mendapatkan data log jaringan yang kemudian digunakan sebagai data uji. Log jaringan yang didapatkan kemudian diekstrak sehingga diperoleh data dengan 84 fitur dan 31167 baris data dengan rincian seperti pada Tabel 4.

Tabel 4 Hasil Ekstraksi Log Jaringan

Menit	**Kelas**	**Baris Data**	**Hasil Ekstraksi**
2	Normal	796	41
1	Serangan	153913	3309
2	Normal	1460	399
1	Serangan	143520	20
2	Normal	143	17
1	Serangan	153605	6454
2	Normal	3335	798
1	Serangan	156429	3819
2	Normal	1638	585
1	Serangan	150243	5297
2	Normal	1526	404
1	Serangan	285687	9509
2	Normal	1573	515

Pelabelan dilakukan secara manual dengan berpatokan pada skema waktu simulasi serangan. Sebanyak 2779 (8.9%) baris data untuk kelas normal dan sebanyak 28388 (91.1%) baris data untuk kelas serangan. Karena jumlah antara data kedua kelas terpaut jauh, maka dilakukan reduksi agar data kedua kelas tersebut menjadi seimbang. Setelah dilakukan reduksi, didapatkan data untuk kelas normal sebanyak 2779 (53.1%) baris data dan data untuk kelas serangan sebanyak 3147 (46.9%) baris data. Kelas normal kemudian diberi label BENIGN dan kelas serangan diberi label DDoS, seperti halnya yang ada pada data latih.

Grafik perbandingan distribusi label BENIGN dan label DDoS pada data latih dan data uji dapat dilihat pada Gambar 7.



Gambar 7 Grafik Dristribusi Label Data Latih dan Data Uji

Dari hasil pengecekan data latih dan data uji, ada beberapa fitur yang tidak perlu untuk proses analisis dan beberapa fitur yang hanya memiliki nilai 0 sehingga sebaiknya dihapus saja. Fitur-fitur tersebut seperti yang dapat dilihat pada Tabel 5.

Tabel 5 Fitur-Fitur yang Tidak Perlu

No	Nama Fitur
1.	flow_id
2.	source_ip
3.	source_port
4.	destination_ip
5.	timestamp
6.	fwd_urg_flags
7.	bwd_urg_flags
8.	cwe_flag_count
9.	fwd_avg_bytes_bulk
10.	fwd_avg_packets_bulk
11.	fwd_avg_bulk_rate
12.	bwd_avg_bytes_bulk
13.	bwd_avg_packets_bulk
14.	bwd_avg_bulk_rate

Selain itu, fitur *flow_bytes* dan *flow_packets* pada data latih terindikasi menyimpan nilai *NaN* dan *Infinity* sedangkan pada data uji tidak ada. Hal ini menyebabkan tipe data dari kedua fitur tersebut menjadi ambigu. Oleh sebab itu tipe datanya perlu diganti menjadi *float64* terlebih dahulu, kemudian mengganti nilai NaN dan *Infinity* dengan nilai 0 (nilai konstan).

Data kemudian dinormalisasikan agar data memiliki rata-rata nol dan varian unit, atribut numerik diekstrak dan diseleksi terlebih dahulu sehingga fitur label tidak ikut distandarisasikan karena merupakan fitur dengan atribut kategorik. Dari proses standarisasi, atribut numerik kemudian berubah seperti yang dapat dilihat pada Gambar 8 dan Gambar 9.

```
sc_dftrain.head()
```

	destination_port	protocol	flow_duration	total_fwd_packets	total_bwd_packets
0	2.327831	-0.412278	-0.515210	-0.186406	-0.210191
1	2.337398	-0.412278	-0.515207	-0.251245	-0.164225
2	2.337449	-0.412278	-0.515209	-0.251245	-0.164225
3	1.891022	-0.412278	-0.515209	-0.251245	-0.164225
4	2.327730	-0.412278	-0.515210	-0.186406	-0.210191

Gambar 8 Hasil Normalisasi Atribut Numerik Pada Data Latih

```
sc_dftest.head()
```

	destination_port	protocol	flow_duration	total_fwd_packets	total_bwd_packets
0	1.984230	-0.078876	-0.695704	-0.054584	0.009255
1	-0.182238	10.768363	-0.661010	-0.053357	-0.134694
2	1.985507	-0.078876	-0.695704	-0.054584	0.009255
3	1.984372	-0.078876	-0.695704	-0.054584	0.009255
4	1.985129	-0.078876	-0.695704	-0.054584	0.009255

Gambar 9 Hasil Normalisasi Atribut Numerik Pada Data Uji

Selain atribut numerik, atribut kategorik juga perlu dinormalisasikan. Atribut kategorik yang dalam hal ini yaitu yang ada pada fitur label. Atribut kategorik diekstrak dan diseleksi terlebih dahulu, kemudian atribut yang ada di fitur label berubah menjadi angka 0 dan 1. Angka 0 merepresentasikan kelas BENIGN, sedangkan angka 1 merepresentasikan kelas DDoS.

Sebelum fitur-fitur diseleksi, diambil beberapa sampel data untuk menambah atribut pada kelas yang kurang sehingga kelas yang ada pada fitur label data latih menjadi sama dan seimbang, yang sebelumnya berjumlah 97718 baris data untuk kelas BENIGN dan 128027 baris data untuk kelas DDoS, menjadi 128027 baris data untuk kedua kelas.

Dari proses seleksi fitur, diperoleh 20 fitur penting yang diurutkan berdasarkan skor seberapa penting fitur tersebut. Fitur-fitur tersebut dapat dilihat pada Tabel 6.

Tabel 6 Fitur Terpilih Hasil Seleksi Fitur

No	Fitur Terpilih	Skor
1.	act_data_pkt_fwd	0.18469
2.	avg_fwd_segment_size	0.12274
3.	fwd_packet_length_max	0.10458
4.	subflow_fwd_bytes	0.09058
5.	bwd_packet_length_std	0.06399
6.	init_win_bytes_forward	0.05179
7.	total_length_of_fwd_packets	0.05094
8.	bwd_packet_length_max	0.04691
9.	subflow_fwd_packets	0.04265
10.	bwd_packet_length_min	0.03666
11.	avg_bwd_segment_size	0.03316
12.	bwd_header_length	0.02401
13.	fwd_iat_min	0.02331
14.	fwd_packet_length_mean	0.02165

15.	bwd_iat_min	0.02105
16.	flow_iat_std	0.01787
17.	bwd_iat_total	0.01131
18.	flow_packets	0.00915
19.	flow_duration	0.00901
20.	packet_length_mean	0.0058

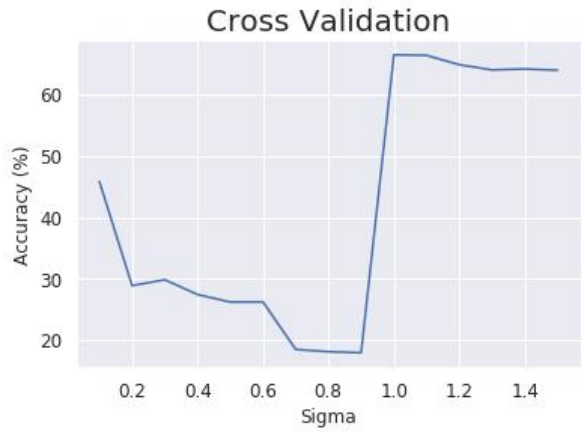
Pelatihan dilakukan dengan mengimplementasikan teknik *Cross Validation* untuk mendapatkan parameter sigma (σ) terbaik. Jumlah iterasi yang diterapkan yaitu sebanyak 15 kali dengan penambahan sigma sebesar 0.1 di setiap iterasi. Setelah mendapatkan sigma terbaik, maka selanjutnya dilakukan pengujian. Percobaan dilakukan dua kali, yaitu pelatihan dan pengujian dengan fitur lengkap serta pelatihan dan pengujian dengan fitur terpilih.

Proses pelatihan dengan fitur lengkap membutuhkan waktu 1 jam 45 menit 6 detik. Tabel 7 menunjukkan hasil dari pelatihan dengan fitur lengkap.

Tabel 7 Hasil Pelatihan dengan Fitur Lengkap

Iterasi	Sigma	Akurasi
1	0.1	45.8%
2	0.2	28.9%
3	0.3	29.8%
4	0.4	27.4%
5	0.5	26.2%
6	0.6	26.2%
7	0.7	18.5%
8	0.8	18.1%
9	0.9	18.0%
10	1.0	66.4%
11	1.1	66.3%
12	1.2	64.8%
13	1.3	64.0%
14	1.4	64.1%
15	1.5	63.9%

Sedangakan hasil pelatihan dengan fitur lengkap dalam bentuk grafik dapat dilihan pada Gambar 10.



Gambar 10 Grafik Hasil Pelatihan dengan Fitur Lengkap

Dari hasil pelatihan yang telah dilakukan, diperoleh parameter nilai sigma atau deviasi standar terbaik yaitu sebesar 1 yang menunjukkan data uji terdistribusi dengan cukup baik dimana titik data uji cukup mirip atau mendekati terhadap data latih pada sigma 1 yang masih terbilang normal. Dengan demikian, sigma 1 digunakan sebagai masukan untuk pengujian. Gambar 11 menunjukkan hasil pengujian dengan fitur terpilih.

```
predictions = grnn.predict(X_test, sigma=1)
print(predictions)
```

```
[0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 1. 0. 1. 0. 1. 1. 0. 0. 0. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1.
```

Gambar 11 Hasil Pengujian dengan Fitur Lengkap

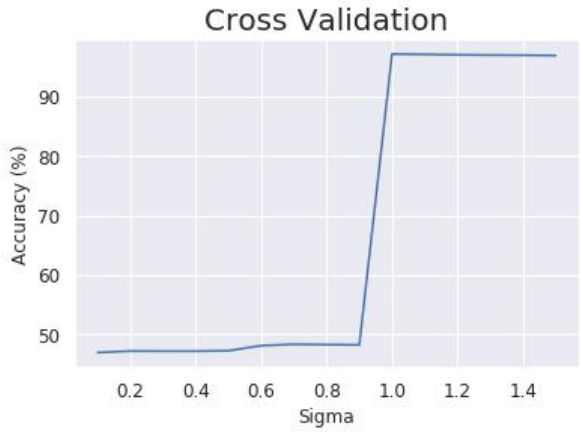
Proses pelatihan dengan fitur terpilih membutuhkan waktu selama 42 menit 27 detik. Adapun hasil dari pelatihan dengan fitur terpilih dapat dilihat pada Tabel 8.

Tabel 8 Hasil Pelatihan dengan Fitur Terpilih

Iterasi	Sigma	Akurasi
1	0.1	46.9%
2	0.2	47.2%
3	0.3	47.1%
4	0.4	47.1%
5	0.5	47.2%
6	0.6	48.1%
7	0.7	48.3%
8	0.8	48.2%
9	0.9	48.2%
10	1.0	97.2%
11	1.1	97.1%
12	1.2	97.1%
13	1.3	97.0%
14	1.4	97.0%
15	1.5	97.0%

.

Sedangakan hasil pelatihan dengan fitur lengkap dalam bentuk grafik dapat dilihan pada Gambar 12.



Gambar 12 Grafik Hasil Pelatihan dengan Fitur Terpilih

Dari hasil pelatihan yang telah dilakukan, diperoleh parameter nilai sigma atau deviasi standar terbaik yaitu sebesar 1 yang menunjukkan data uji terdistribusi dengan cukup baik dimana titik data uji cukup mirip atau mendekati terhadap data latih pada sigma 1 yang masih terbilang normal. Dengan demikian, sigma 1 digunakan sebagai masukan untuk pengujian. Gambar 13 menunjukkan hasil pengujian dengan fitur terpilih.

```
predictions2 = grnn2.predict(X_test_fs, sigma=1)
print(predictions2)
```

```
[0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1.
 1. 1. 1. 1. 1. 0. 0. 0. 1. 1. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
```

Gambar 13 Hasil Pengujian dengan Fitur Terpilih

Hasil evaluasi yang dilakukan dengan fitur lengkap, diperoleh *accuracy* sebesar 66.41%; *precision* sebesar 73,85%; *recall* sebesar 64,52%; dan *f1-score* sebesar 61,89% seperti yang dapat dilihat pada Gambar 14.

```
Model Accuracy:
 0.6641916976037799

Confusion matrix:
 [[ 947 1832]
 [ 158 2989]]

Classification report:
              precision    recall  f1-score   support

           0       0.86      0.34      0.49      2779
           1       0.62      0.95      0.75      3147

    accuracy                           0.66      5926
   macro avg       0.74      0.65      0.62      5926
weighted avg       0.73      0.66      0.63      5926
```

Gambar 14 Hasil Evaluasi Model dengan Fitur Lengkap

Sedangkan hasil evaluasi yang dilakukan dengan fitur terpilih, diperoleh *accuracy* sebesar 97,21%; *precision* sebesar 97,21%; *recall* sebesar 97,19%; dan *f1-score* sebesar 97,2% seperti yang dapat dilihat pada Gambar 15.

```
Model Accuracy:
 0.9721565980425245

Confusion matrix:
 [[2690   89]
 [  76 3071]]

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.97      0.97      2779
           1       0.97      0.98      0.97      3147

    accuracy                           0.97      5926
   macro avg       0.97      0.97      0.97      5926
weighted avg       0.97      0.97      0.97      5926
```

Gambar 15 Hasil Evaluasi Model dengan Fitur Terpilih

V. KESIMPULAN

Berdasarkan dari hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa *General Regression Neural Network* mampu mendeteksi serangan DDoS dengan baik. Analisis dilakukan dengan menggunakan data latih dari set data terbaru untuk deteksi intrusi yaitu CICIDS2017 yang merupakan pengembangan dari set data yang telah ada sebelumnya. Sedangkat untuk data uji diperoleh dari hasil simulasi serangan DDoS ke server web yang diekstrak menggunakan CICFlowMeter 4.0 sehingga menghasilkan data dengan format yang sama seperti data latih.

Dua percobaan berbeda dilakukan pada penelitian ini, pada percobaan pertama, pelatihan dan pengujian dilakukan dengan menggunakan semua fitur yang ada pada data latih dan data uji yaitu sebanyak 69 fitur. Percobaan ini membutuhkan waktu selama 1 jam 45 menit 6 detik untuk pelatihan. Dari pelatihan tersebut diperoleh sigma terbaik yaitu 1 yang kemudian dijadikan masukan untuk pengujian dan evaluasi. Dari hasil evaluasi ini, diperoleh *accuracy* sebesar 66.41%; *precision* sebesar 73,85%; *recall* sebesar 64,52%; dan *f1-score* sebesar 61,89%.

Sementara itu pada percobaan kedua, pelatihan dan pengujian dilakukan dengan memanfaatkan sebagian fitur yang telah diseleksi menggunakan algoritma *Random Forest*. Dari hasil seleksi fitur, diperoleh sebanyak 20 fitur terpilih. Percobaan ini membutuhkan waktu selama 42 menit 27 detik untuk pelatihan. Dari pelatihan tersebut diperoleh sigma terbaik yaitu 1 yang kemudian dijadikan masukan untuk pengujian dan evaluasi. Dari hasil evaluasi ini, diperoleh *accuracy* sebesar 97,21%; *precision* sebesar 97,21%; *recall* sebesar 97,19%; dan *f1-score* sebesar 97,2%.

Dari kedua percobaan yang telah dilakukan, percobaan kedua menghasilkan *accuracy*, *precision*, *recall*, dan *f1-score* yang jauh lebih baik dari percobaan pertama. Dapat disimpulkan bahwa optimalisasi parameter sigma merupakan hal yang sangat penting pada saat evaluasi model. Hal ini disebabkan karena sigma secara langsung mampu mempengaruhi sebaran

data pada data, apabila sigma terlalu besar maka hasil yang diperoleh pun akan tidak optimal. Seleksi fitur juga memiliki peran penting dalam memperoleh *accuracy*, *precision*, *recall*, dan *f1-score* yang optimal. Hal ini disebabkan karena fitur-fitur yang terpilih adalah fitur yang memiliki pengaruh penting terhadap *y* atau variabel dependen. Selain itu seleksi fitur juga dapat memangkas waktu komputasi menjadi lebih singkat.

REFERENSI

[1] *Web Hacking Incident Database*, 2017.

[2] Mehic, M., Slachta, J., Voznak, M. 2016. *Whispering through DDoS attack*. Perspectives in Science Vol 7, 95-100.

[3] Kato, K. & Klyuev, V. 2014. *Large-scale network packet analysis for intelligent DDoS attack detection development*. 9th Inernational Conference for Internet Technology and Secure Transansactions. ICITST 2014, 360-365.

[4] Goodin, D. 2018. *US service provider survives the biggest recorded DDoS in history,* [online], (https://arstechnica.com/ information-technology/2018/03/us-service-provide r-survives- the-biggest-recorded-ddos-in-history/, diakses Mei 2018)

[5] Specht, D. F. 1991. A *General Regression Neural Network*. IEEE Transactions On Neural Networks. Vol. 2 . No. 6. November 1991.

[6] Rizzo, R., Fiannaca, A., La Rosa, M., Urso, A. 2015. *The General Regression Neural Network to Classify Barcode and mini-barcode DNA.* CIBB 2014, LNCS 8623, pp. 142–155.

[7] DTERG Predictive Modelling Software, [online], (https://www.dtreg.com/solution/view/22, diakses Desember 2019).

[8] Canadian Institute for Cybersecurity, [online], (http://www.netflowmeter.ca/, diakses Maret 2019).

[9] Panigrahi, R. & Borah, S. 2018. *A Detailed Analysis of CICIDS2017 Dataset for Designing Intrusion Detection Systems.* International Journal of Engineering & Technology, 7 (3.24) : 479-482.