PENERAPAN RECURRENT NEURAL NETWORK DENGAN LONG SHORT TERM MEMORY UNTUK MENDETEKSI MALWARE BOTNET BERDASARKAN FITUR NETWORK TRAFFIC

TUGAS AKHIR

Diajukan sebagai syarat untuk menyelesaikan
Program Studi Strata-1 Informatika

Disusun Oleh:
Joshua Allen
1115011

INSTITUT
TEKNOLOGI
HARAPAN
BANGSA

Veritas vos liberabit

PROGRAM STUDI INFORMATIKA
INSTITUT TEKNOLOGI HARAPAN BANGSA
BANDUNG
2020

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Seiring berkembangnya pemanfaatan komputer yang terkoneksi dengan internet, *botnet* telah menjadi salah satu ancaman dalam keamanan jaringan [1]. *Botnet* dapat didefinisikan sebagai kumpulan dari komputer yang telah terinfeksi untuk dapat dikontrol jarak jauh dengan tujuan mengeksekusi serangan yang terkoordinasi melalui sebuah *software malicious*. *Botnet* diciptakan oleh penyerang untuk menjalankan *Distributed Denial of Service (DDoS)* terhadap *website* berskala besar, mencuri data penting dari komputer, mengirimkan *bulk email* berisi iklan, atau mengirimkan *click fraud* [1]. Komputer yang telah terinfeksi dan menjadi bagian dari sebuah *botnet* disebut *bot client*.

Berbeda dengan virus atau *malware* konvensional seperti *trojan, worm*, dan virus lainnya, *botnet malware* memiliki kelebihan dapat berinteraksi dengan *botmaster* melalui *Command and Control (C&C) communication channel*. *Bot* tergabung dalam *C&C channel* melalui banyak varian protokol komunikasi seperti *IRC*, *HTTP/HTTPS*, *P2P Protocol*, dan lainnya [2]. Di akhir tahun 2017, Spamhaus Malware Lab telah berhasil mengidentifikasikan lebih dari 9500 *botnet C&C Server* pada 1122 jaringan yang berbeda. Angka ini telah meningkat sebanyak 40% dibandingkan tahun 2016 dan 90% dibandingkan tahun 2014 [9].

Peningkatan jumlah *botnet* yang tergolong tinggi setiap tahunnya, menuntut adanya perkembangan metode serta pendekatan yang digunakan untuk mendeteksi *botnet*. Di penelitian lampau, deteksi *botnet* dengan algoritma *Particle Swarm Optimization* (PSO) dan *Support Vector Machine* (SVM) telah mencapai akurasi 87% *True Positive Rate* [10]. Namun metode ini memiliki kelemahan yaitu algoritma SVM memerlukan data yang sudah terstruktur dengan panjang vektor dan jumlah fitur tertentu, sehingga dinilai belum dapat mendeteksi *botnet* dengan fitur dari *network traffic* yang luas dan bersifat fleksibel. Penelitian selanjutnya berhasil menerapkan *Artificial Neural Network (ANN)* terhadap *network traffic behaviour* dengan akurasi 95% *True Positive Rate* terhadap *malicious* [11]. Kelemahan *Artificial Neural Network* dalam mendeteksi *malware botnet* dikarenakan oleh aktivitas *botnet* yang hanya terjadi pada satu interval waktu tertentu yang dapat terpotong-potong dan akan kembali *idle* apabila tidak menerima perintah dari *C&C server* memungkinkan tidak terdeteksinya *botnet* karena masukan yang tidak bergantung satu sama lain.

Penelitian selanjutnya menerapkan *Recurrent Neural Network (RNN)* untuk mendeteksi *botnet* berdasarkan sekuensial waktu. Penelitian tersebut menyebutkan bahwa metode ini memiliki kekurangan, yaitu *vanishing gradient problem*. *Vanishin gradient problem* adalah mengecilnya nilai *gradient* untuk melakukan *update* terhadap *bobot* saat melakukan

back propagation sehingga proses belajar akan membutuhkan waktu yang sangat lama atau tidak bekerja sama sekali [5]. Untuk mengatasi masalah yang dimiliki oleh *Recurrent Neural Network*, penelitian menggunakan *Long Short Term Memory* pun dilakukan. Long Short Term Memory adalah salah satu varian dari *Recurrent Neural Network* dimana sebuah unit *Long Short Term Memory* terdiri dari sebuah *cell*, *input gate*, *output gate*, dan *forget gate*. *Cell* atau yang akan kita sebut memori mengingat nilai yang berubah-ubah sesuai interval waktu dan tiga *gate* lain yang berfungsi untuk mengatur jalannya informasi ke dalam dan ke luar sel. *Long Short Term Memory* mampu menanggulangi perbedaan interval waktu yang besar dan sebagai solusi untuk *vanishing gradient problem* yang dimiliki oleh *Recurrent Neural Network*. Penelitian menggunakan *Long Short Term Memory* untuk mendeteksi *malware botnet* berhasil dilakukan dengan akurasi 80.9% dengan menggunakan 3 fitur yaitu *flow size*, *flow duration*, dan *flow periodicity*[3].

Penelitian ini akan menguji metode *Recurrent Neural Network* dengan *Long Short Term Memory* dengan pendekatan yang berbeda yaitu menggunakan beberapa set fitur yang berbeda dan konfigurasi model *Long Short Term Memory* yang beragam. Tujuan penelitian ini adalah menganalisis fitur dan menentukan konfigurasi model *Long Short Term Memory* untuk mendeteksi *malware botnet* dengan *dataset* yang berbeda dengan harapan dapat membentuk sebuah model *Long Short Term Memory* yang dapat diaplikasikan dan mencapai akurasi yang baik dalam mendeteksi aktivitas *malware botnet*.

1.2 Rumusan Masalah

Berikut ini merupakan rumusan masalah yang menyangkut ide pokok dari setiap masalah yang akan dibahas dan dipecahkan melalui penelitian ini:

1. Apa saja fitur *network traffic* yang dapat digunakan untuk mendeteksi aktivitas *malware botnet*?
2. Bagaimana konfigurasi model *Long Short Term Memory* untuk mencapai akurasi tertinggi dalam mendeteksi *malware botnet* berdasarkan fitur *network traffic*?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah diatas, maka dapat dideskripsikan tujuan dari penelitian ini, yaitu :

1. Menentukan fitur-fitur *network traffic* yang dapat digunakan untuk mendeteksi aktivitas *malware botnet*.
2. Mengimplementasikan dan menganalisis konfigurasi model *Long Short Term Memory* untuk mencapai akurasi tertinggi dalam mendeteksi *malware botnet* berdasarkan fitur *network traffic*.

1.4 Batasan Masalah

Agar penelitian ini lebih fokus dan terarah, maka penulis akan memberikan beberapa batasan masalah sebagai berikut:

1. Data yang digunakan berasal dari ISOT Dataset 2010

1.5 Kontribusi Penelitian

Kontribusi dari penelitian ini adalah implementasi metode *Long Short Term Memory* terhadap fitur *network traffic* untuk mendeteksi *malware botnet* pada keamanan jaringan.

1.6 Metodologi Penelitian

Berikut ini merupakan metodologi penelitian yang dilakukan dalam penelitian ini:

1. Tinjauan Pustaka
   Mengumpulkan data dan informasi dengan menggunakan referensi pustaka yang berasal dari berbagai media dan sumber (artikel, buku, literatur, jurnal, *paper*, tugas akhir, dan thesis) yang berhubungan dengan penelitian yang dilakukan dan metode yang digunakan.
2. Data *Sampling*
   Pada tahap ini penulis menggunakan *dataset* yang diperoleh dari internet yang akan digunakan untuk proses penelitian.
3. Analisis dan Perancangan
   Menganalisis dan merancang sistem yang akan dibuat berdasarkan data *sampling* yang sudah diperoleh sebelumnya.
4. Implementasi
   Mengimplementasikan hasil perancangan yang telah dibuat sebelumnya dalam bentuk sistem klasifikasi *malware botnet*.
5. Pengujian
   Melakukan pengujian terhadap sistem yang telah dibuat dan menghitung nilai evaluasi dari hasil klasifikasi *malware botnet*.

1.7 Sistematika Penulisan

Laporan penelitian ini disusun berdasarkan sistematika penulisan berikut ini:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, metodologi penelitian, dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini menjelaskan teori mengenai tinjauan pustaka, tinjauan studi, dan tinjauan objek.

BAB III ANALISIS DAN PERANCANGAN

Bab ini menjelaskan analisa terhadap masalah, solusi, dan perancangan sistem.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Bab ini menjelaskan proses implementasi dan pengujian terhadap aplikasi yang dibuat serta pengukuran nilai evaluasi dari hasil aplikasi.

BAB V KESIMPULAN DAN SARAN

Bab ini menjelaskan kesimpulan dan saran untuk mendukung perbaikan sistem aplikasi ini.

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Pada bagian ini akan dijelaskan mengenai teori-teori terkait yang akan digunakan dalam penelitian ini.

2.1.1 Pembelajaran Mesin

Machine Learning atau pembelajaran mesin merupakan salah satu cabang dari kecerdasan buatan. Pembelajaran mesin berusaha membuat komputer belajar tanpa rangkaian algoritmik secara langsung. Proses pembelajaran mesin terdiri dari tahap pelatihan dan pengujian. Tahap pelatihan digunakan untuk membentuk model, sedangkan tahap pengujian digunakan untuk memperoleh hasil menggunakan model yang sudah dibuat pada tahap pelatihan.

Pelatihan merupakan proses menjalankan perintah program komputer untuk dapat mengoptimalkan parameter model dari data yang diperoleh sebelumnya. Model dalam pembelajaran mesin dapat dibentuk dari beberapa parameter. Sedangkan bentuk model dapat berupa prediktif, yaitu untuk memprediksi hasil dari kemiripan data sebelumnya, deskriptif untuk memperoleh pengetahuan dari data, ataupun keduanya.

Secara umum, pembelajaran mesin dapat dibagi menjadi empat teknik berdasarkan cara mesin memahami data, yaitu [15].

1. *Supervised Learning*, yaitu teknik dalam pembelajaran mesin yang memerlukan contoh beserta hasil yang diinginkan dalam jumlah data yang besar. *Supervised Learning* menghasilkan model yang akan memberikan hasil berupa klasifikasi atau regresi.
2. *Unsupervised Learning*, yaitu teknik dalam pembelajaran mesin dimana data yang digunakan tidak memiliki label. Salah satu penerapan *unsupervised learning* adalah *clustering*, yaitu pengelompokan data dengan karakteristik yang serupa.
3. *Semi-supervised Learning*, yaitu teknik dalam pembelajaran mesin dengan menggabungkan teknik *supervised* dan teknik *unsupervised* untuk saling melengkapi kelemahan masing-masing teknik.
4. *Reinforcement Learning*, yaitu teknik dalam pembelajaran mesin yang melakukan tindakan secara langsung serta menerima dampaknya untuk menyelesaikan persoalan.

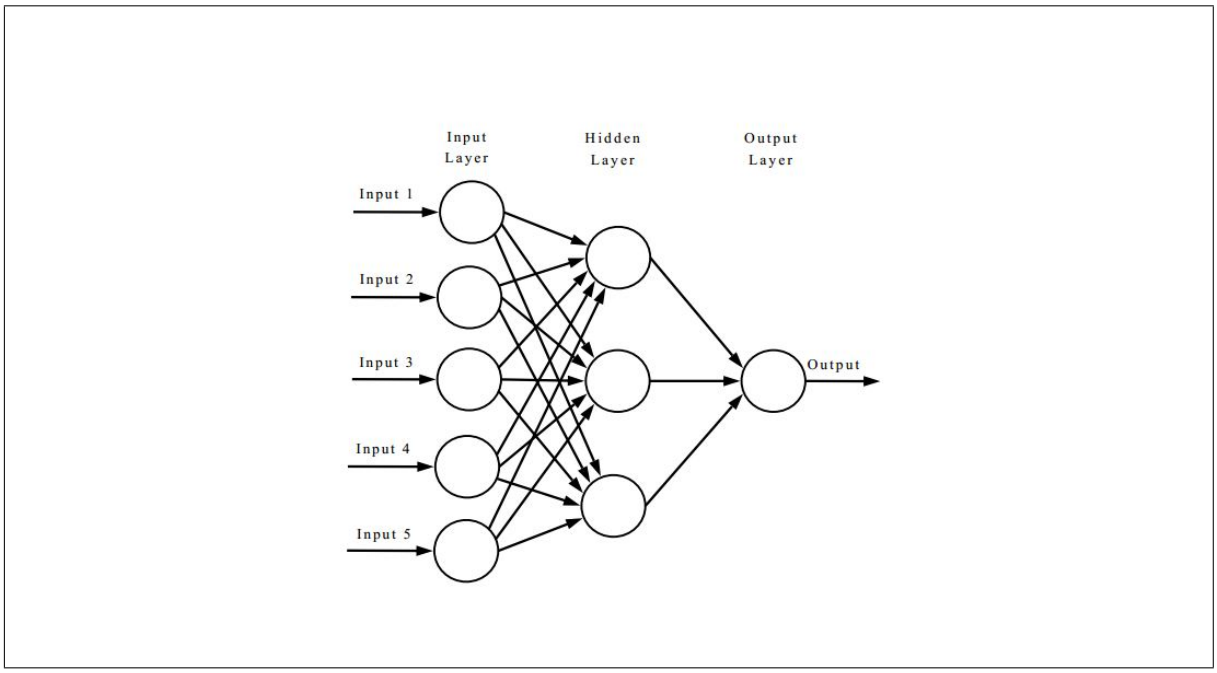
2.1.2 *Supervised Learning*

Supervised learning merupakan teknik pembelajaran mesin untuk memahami data dari kombinasi beberapa fitur dengan nilai keluaran. Tujuan utama teknik ini adalah untuk membentuk model yang dapat digunakan untuk memperoleh nilai prediksi dengan menggunakan data baru. Teknik *supervised learning* dapat dibagi menjadi dua, yaitu klasifikasi dan regresi. Klasifikasi akan menghasilkan keluaran dalam bentuk label kelas yang sesuai dari data masukan yang diterima. Sedangkan, regresi digunakan apabila nilai keluaran dari data merupakan bilangan riil [15].

Neural network merupakan salah satu metode yang dapat digunakan untuk pembelajaran mesin dengan teknik *supervised learning*. Teknik *supervised learning* dengan menggunakan *neural network* dalam bidang keamanan jaringan banyak digunakan untuk mendeteksi *spam*, *botnet*, dan deteksi pola malware.

2.1.3 *Artificial Neural Network* (ANN)

ANN adalah sebuah aplikasi dari pembelajaran mesin menggunakan metode *supervised learning* yang meniru kinerja otak manusia dalam menyelesaikan sebuah masalah. ANN bekerja dengan mengambil beberapa masukan kemudian diproses dalam beberapa lapisan *neuron* atau *node* di dalam beberapa lapisan tersembunyi dan mengembalikan hasil seperti digambarkan pada Gambar 2.1 [7].



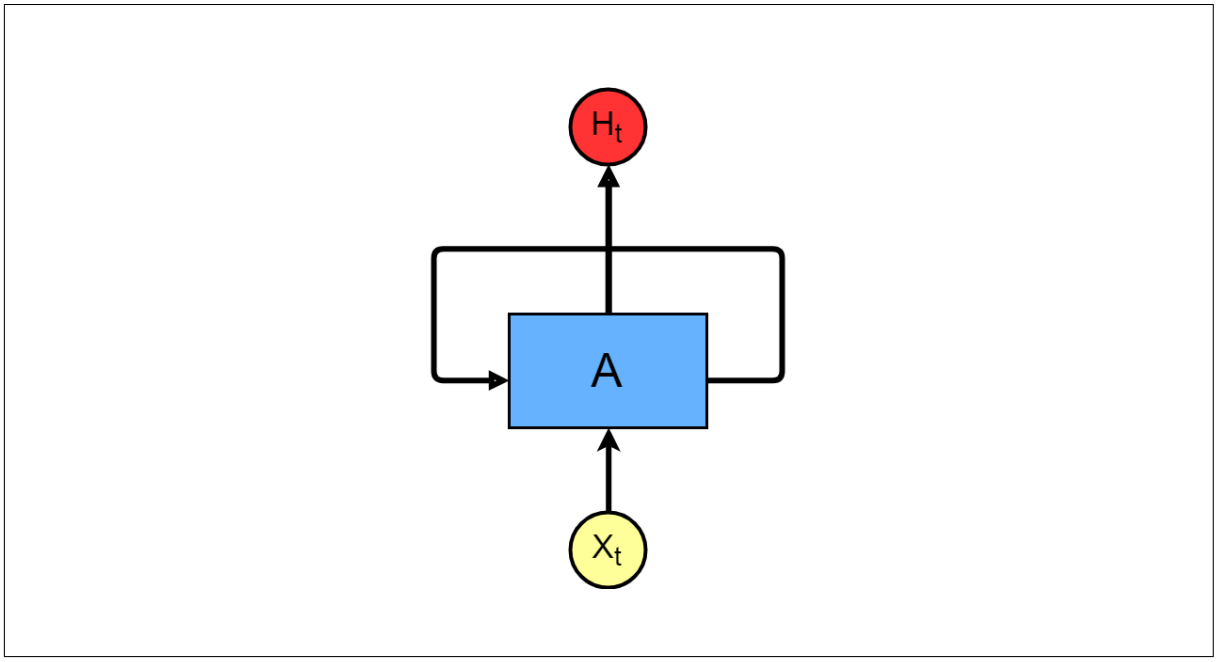
Gambar 2.1 *Artificial Neural Network Structure*

Satu kali proses dari pengambilan masukan sampai diperoleh keluaran dalam ANN dinamakan proses *forward propagation*. Keluaran tersebut kemudian dibandingkan dengan hasil sesungguhnya atau hasil sebenarnya yang diinginkan, seberapa besar *error* yang dihasilkan dari proses *forward propagation*. Langkah selanjutnya adalah meminimalisir error

yang terjadi dengan memperbaiki *neuron* yang paling berpengaruh dalam menghasilkan jumlah *error*. Meminimalisir *error* dilakukan dengan dengan melangkah mundur dari lapisan-lapisan tersembunyi dalam sistem ANN dan menemukan neuron penyebab *error* tersebut, proses ini dinamakan dengan *backward propagation* [7].

2.1.4 *Recurrent Neural Network* (RNN)

RNN adalah salah satu kelas dari ANN dimana koneksi antar *node* membentuk sebuah grafik yang bersambung [18]. RNN melakukan fungsi yang sama untuk setiap masukan data. Setelah menghasilkan keluaran data, data tersebut kemudian dikirim kembali pada unit berikutnya dari *recurrent network*. Dalam membuat keputusan, RNN mempertimbangkan masukan titik data saat ini dan keluaran dari unit sebelumnya. Hal inilah yang membedakan RNN dari ANN, masukan dan keluaran pada ANN tidak bergantung satu sama lain sedangkan RNN memiliki sebuah memori berisikan hasil perhitungan informasi yang dihasilkan sebelumnya.

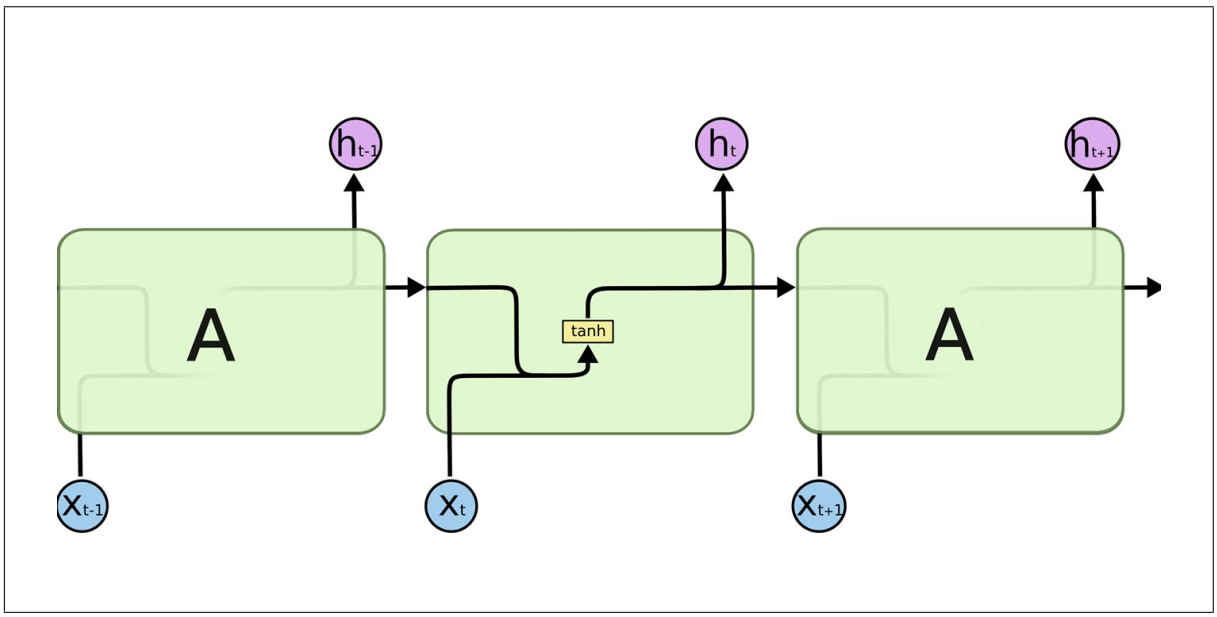


Gambar 2.2 *Recurrent Neural Network Loop*

Keterangan:

1. A = unit RNN
2. X_t = masukan pada *time step* t
3. H_t = keluaran pada *time step* t

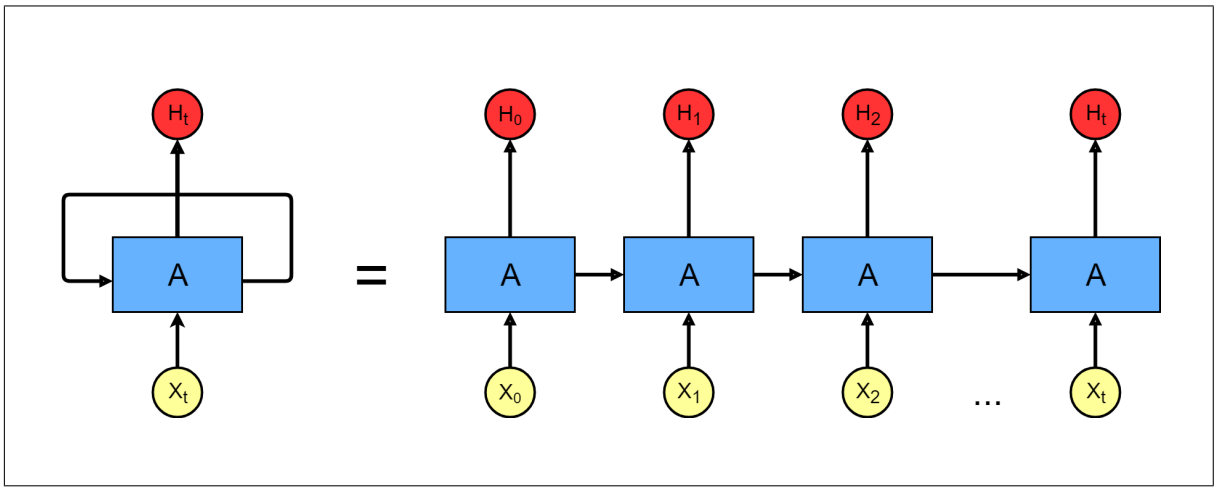
Gambar 2.2 menunjukkan sebuah proses RNN dimana unit A menerima masukan X_t dan menghasilkan keluaran H_t. Dalam gambar 2.3 ditunjukkan bahwa *loop* yang diciptakan oleh unit A memungkinkan informasi untuk diteruskan dari satu unit ke unit berikutnya.



Gambar 2.3 *Recurrent Neural Network*

Gambar 2.4 menunjukkan RNN akan memproses keluaran dari unit RNN sebelumnya sebagai masukan. Sifat dasar RNN yang sekuensial ini menunjukkan bahwa model RNN memiliki arsitektur yang cocok untuk data berbentuk *sequence* atau *list* [8].

Proses RNN dapat dilihat dalam bagan berikut.



Gambar 2.4 *Unrolled Recurrent Neural Network*

Pertama, unit 0 akan memproses masukan X_0 dan menghasilkan keluaran H_0. Setelah perhitungan unit 0 selesai, keluaran unit 0 H_0 dan masukan unit 1 X_1 keduanya akan menjadi masukan untuk unit 1 dan demikian selanjutnya. Dengan demikian unit RNN akan tetap mengingat konteks dari hasil pembelajaran sebelumnya.

Berikut persamaan 2 . 1 untuk menghitung memori layer pada RNN.

$$H_t = f(H_{t-1}, X_t) \quad (2.1)$$

Fungsi pada persamaan 2 . 1 dijelaskan lebih rinci pada persamaan 2 . 2, yang menunjukkan bahwa masukan pada titik data saat ini akan diterapkan fungsi aktivasi *Hyperbolic Tangent* bersama dengan nilai memori dari unit sebelumnya.

$$H_t = tanhf(W_{hh}H_{t-1} + W_{xh}X_t) \quad (2.2)$$

Keterangan:

1. W = Bobot
2. H = Memori
3. W_{hh} = Bobot memori sebelumnya
4. W_{hx} = Bobot masukan
5. $tanh$ = Fungsi aktivasi *Hyperbolic Tangent*

Berikut persamaan 2 . 3 untuk menghitung keluaran dari sistem RNN dengan mengaktivasi nilai memori terhadap bobot keluaran.

$$Y_t = W_{hy}H_t \quad (2.3)$$

Keterangan:

1. Y_t = Keluaran
2. W_{hy} = Bobot keluaran
3. H = Memori

2.1.5 Fungsi Aktivasi

Fungsi aktivasi dalam *neural network* digunakan untuk mengubah data masukan yang telah diberikan bobot menjadi *neuron* tersembunyi. Berikut merupakan beberapa fungsi aktivasi pada *neural network* yang akan digunakan dalam perhitungan LSTM pada penelitian ini:

1. *Sigmoid Function*
   Sigmoid function merupakan fungsi aktivasi yang menghasilkan nilai dengan skala 0 sampai dengan 1. Kelebihan dari fungsi aktivasi sigmoid yang menghasilkan nilai antara 0 dan 1 ini cocok digunakan untuk merepresentasikan kemungkinan terjadinya suatu kondisi. Berikut merupakan persamaan 2 . 4 untuk *sigmoid function*.

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2 . 4}$$

   Keterangan:
   (a) x = masukan bilangan *real*

2. *Hyperbolic Tangent Function*
   Hyperbolic tangent function merupakan fungsi aktivasi yang menghasilkan nilai dengan skala -1 sampai dengan 1. Fungsi aktivasi ini bersifat non-linear dan kelebihannya dibandingkan *sigmoid function* adalah kemampuannya menghasilkan nilai negatif sampai -1. Berikut merupakan persamaan 2 . 5 untuk *hyperbolic tangent function*.

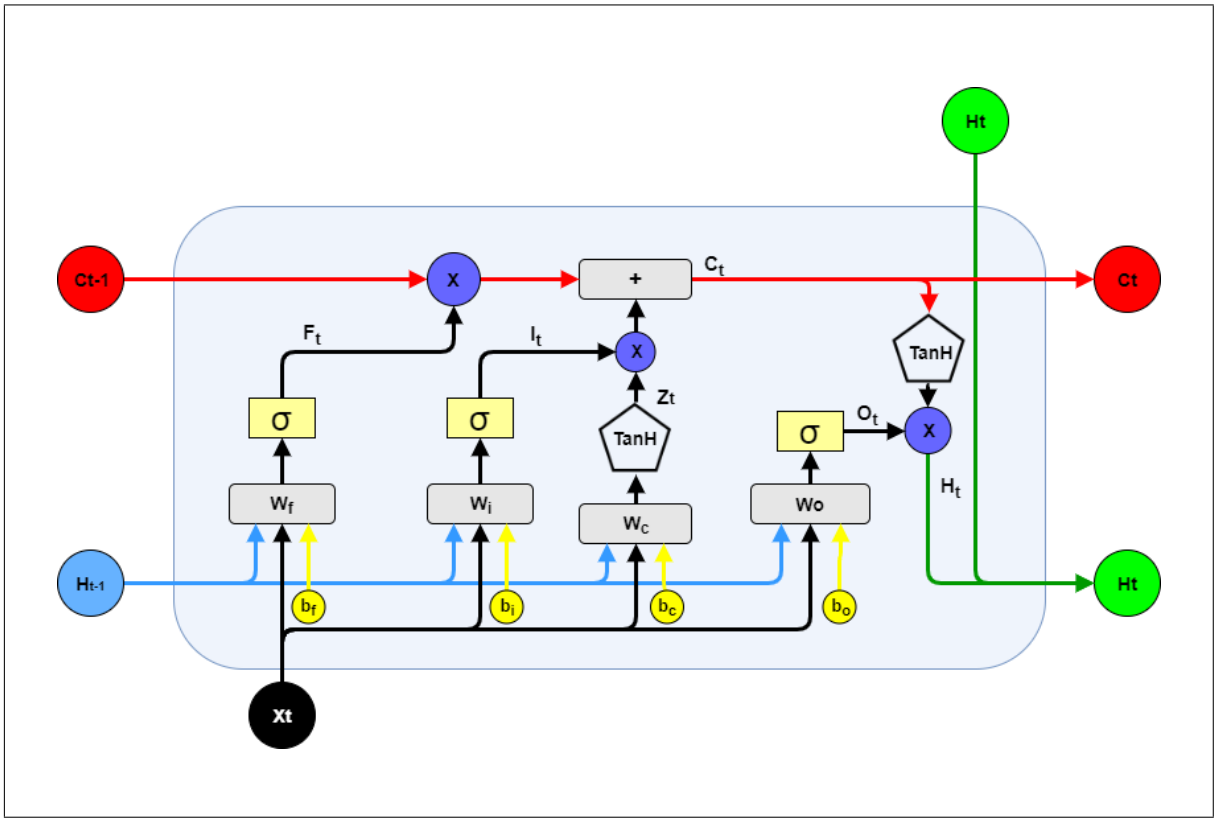
$$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2 . 5}$$

   Keterangan:
   (a) x = masukan bilangan *real*

2.1.6 *Long Short Term Memory* (LSTM)

LSTM adalah salah satu variasi arsitektur RNN yang didesain untuk mampu melakukan pembelajaran jangka panjang dengan memanfaatkan sebuah unit memori yang disebut *cell* [10]. Arsitektur yang dimiliki oleh LSTM memungkinkan informasi dari memori sebuah unit diteruskan tidak hanya pada unit berikutnya, melainkan dapat terus memberikan kontribusi pada unit-unit selanjutnya. LSTM didesain untuk mengatasi *vanishing gradient problem* yang terdapat pada RNN dengan memiliki *Constant Error Carousel* yang

memungkinkan *error* untuk melakukan *backpropagation* tanpa terjadi *vanishing gradient* atau disebut *backpropagation through time* [8].



Gambar 2.5 *LSTM Unit*

Keterangan:

1. X_t = masukan pada *time step* t
2. C_t = memori pada *time step* t
3. C_{t-1} = memori pada *time step* t-1
4. H_t = keluaran pada *time step* t
5. H_{t-1} = keluaran pada *time step* t-1
6. F_t = nilai aktivasi *forget gate*
7. I_t = nilai aktivasi *input gate*
8. Z_t = nilai aktivasi *memory gate*
9. O_t = nilai aktivasi *output gate*
10. σ = fungsi aktivasi sigmoid
11. *TanH* = fungsi aktivasi *hyperbolic tangent*

12. W_f = bobot *forget gate*
13. W_i = bobot *input gate*
14. W_c = bobot *memory gate*
15. W_o = bobot *output gate*
16. b_f = bias *forget gate*
17. b_i = bias *input gate*
18. b_c = bias *memory gate*
19. b_o = bias *output gate*
20. X = fungsi perkalian

Unit LSTM terbagi menjadi 4 *gate* yang berinteraksi untuk menambahkan atau membuang informasi ke dalam memori setiap unit. Gambar 2.5 menjelaskan setiap unit pada jaringan LSTM menerima 2 masukan, yaitu X_t masukan unit tersebut dan h_{t-1} keluaran dari unit sebelumnya Langkah pertama, LSTM akan memutuskan seberapa penting sebuah informasi untuk disimpan. Langkah ini dilakukan oleh *layer sigmoid "forget gate"* dengan persamaan 2 . 6 berikut :

$$F_t = \sigma(W_f.[h_{t-1}, X_t] + b_f) \qquad (2\ .\ 6)$$

Keterangan:

1. F_t = nilai aktivasi *forget gate*
2. σ = fungsi aktivasi *sigmoid*
3. h_{t-1} = keluaran dari unit sebelumnya
4. W_f = bobot *forget gate*
5. X_t = *input* pada *time stamp* t
6. b_f = bias *forget gate*

Fungsi aktivasi *sigmoid* akan menggunakan nilai x_t dan h_{t-1} menghasilkan nilai antara 0 dan 1, dimana hasil 1 menunjukkan bahwa nilai memori akan disimpan seluruhnya sedangkan 0

dihapus seluruhnya.

$$i_t = \sigma(W_i.[h_{t-1}, X_t] + b_i) \quad (2.7)$$

$$z_t = tanh(W_c.[h_{t-1}, X_t] + b_c) \quad (2.8)$$

$$c_t = f_t.c_{t-1} + i_t.z_t \quad (2.9)$$

Keterangan:

1. i_t = nilai aktivasi *input gate*
2. σ = fungsi aktivasi sigmoid
3. W_i = bobot *input gate*
4. h_{t-1} = keluaran dari unit sebelumnya
5. X_t = masukan pada *time stamp* t
6. b_i = bias *input gate*
7. z_t = nilai aktivasi *memory gate*
8. tanh = fungsi aktivasi *hyperbolic tangent*
9. W_c = bobot *memory gate*
10. b_c = bias *memory gate*
11. c_t = memori sel pada *time stamp* t
12. F_t = nilai aktivasi *forget gate*
13. c_{t-1} = memori sel dari unit sebelumnya
14. i_t = nilai aktivasi *input gate*

Langkah berikutnya adalah melakukan perubahan terhadap nilai dari memori unit yang terbagi menjadi 3 tahap. Pertama, *input gate* akan menentukan berapa besar informasi dengan melakukan fungsi aktivasi *sigmoid* terhadap masukan unit ini yang akan dijadikan memori dengan menggunakan persamaan 2 . 7. Tahap kedua, *memory gate* akan melakukan fungsi

aktivasi *hyperbolic tangent* terhadap masukan unit ini untuk menghasilkan nilai memori yang akan digunakan dengan persamaan 2 . 8. Tahap terakhir adalah menghasilkan nilai memori yang diperoleh dari penjumlahan dari hasil perkalian *forget gate* terhadap nilai memori unit sebelumnya dengan hasil perkalian *input gate* dan *memory gate* menghasilkan nilai memori unit saat ini yang akan digunakan untuk perhitungan selanjutnya (Persamaan 2 . 9).

$$o_t = \sigma(W_o.[h_{t-1}, X_t] + b_o) \quad (2\ .\ 10)$$

$$h_t = o_t * tanh(ct) \quad (2\ .\ 11)$$

Keterangan:

1. o_t = nilai aktivasi *output gate*
2. σ = fungsi aktivasi sigmoid
3. h_{t-1} = keluaran dari unit sebelumnya
4. w_f = bobot *output gate*
5. x_t = masukan pada *time stamp* t
6. b_f = bias *output gate*
7. h_t = keluaran pada *time stamp* t
8. c_{t-1} = memori sel dari unit sebelumnya
9. tanh = fungsi aktivasi *hyperbolic tangent*
10. c_t = memori sel pada *time stamp* t

Langkah terakhir adalah menghasilkan keluaran h_t untuk unit saat ini (t). Keluaran yang dihasilkan merupakan nilai dari memori sel yang telah disaring. Pertama, *output gate* akan menentukan jumlah informasi yang akan dijadikan keluaran h_t dengan menggunakan fungsi aktivasi *sigmoid* terhadap masukan unit saat ini dan keluaran dari unit sebelumnya (Persamaan 2 . 10). Kemudian nilai dari *ouput gate* akan dikalikan dengan hasil fungsi aktivasi *hyperbolic tangent* terhadap nilai memori sesuai persamaan 2 . 11, menghasilkan keluaran untuk unit saat ini [4].

Demikian diperoleh hasil keluaran dan memori untuk satu unit LSTM yang akan digunakan oleh unit LSTM berikutnya. Dengan struktur arsitektur ini, maka nilai dari memori

unit akan terus disimpan dan mempengaruhi pengambilan keputusan untuk unit selanjutnya sehingga dapat mengatasi ketidakmampuan RNN dalam mengolah data dengan rentang yang besar.

2.1.7 LSTM *Networks Backward Propagation Through Time*

Backpropagation merupakan salah satu proses belajar dalam perhitungan LSTM dimana proses ini akan menghitung nilai *error* dari keluaran dibandingkan dengan nilai sesungguhnya, kemudian nilai kesalahan ini akan dibagikan sesuai dengan keterlibatan setiap unit LSTM dalam menghasilkan nilai ini. Dengan struktur dalam sebuah unit *LSTM Network* yang lebih kompleks dari unit RNN biasa, maka perhitungan setiap unit untuk mendapatkan perubahan nilai bobot yang digunakan harus melibatkan semua proses aktivasi yang terjadi dimulai dari perhitungan paling akhir yaitu keluaran sampai perhitungan pertama yaitu masukan pada sebuah unit LSTM yang disebut *Backward Propagation Through Time* (BPTT). Masing-masing dari perhitungan BPTT ini akan menghasilkan nilai perubahan bobot yang terlibat di setiap perhitungan yang terjadi setiap langkahnya [17].

Berikut adalah perhitungan yang terlibat dalam proses BPTT dalam LSTM yang menghitung nilai perbedaan (delta) dari semua fungsi aktivasi yang terlibat dalam perhitungan pada LSTM, yaitu aktivasi *input gate*, *forget gate*, *memory gate*, nilai memori, *output gate*, dan nilai keluaran akhir.

Tahap pertama adalah menghitung selisih nilai dari hasil keluaran unit LSTM yang sedang diproses saat ini mempertimbangkan perbedaan nilai keluaran dengan hasil sesungguhnya dan pengaruh keluaran terhadap selisih aktivasi *gate* unit berikutnya. Berikut persamaan 2 . 12 untuk menghitung selisih nilai keluaran.

$$\delta y^t = \Delta^t + W_{hc}\delta z^{t+1} + W_{hi}\delta i^{t+1} + W_{hf}\delta f^{t+1} + W_{ho}\delta o^{t+1} \qquad (2\ .\ 12)$$

Berikut keterangan dari persamaan 2 . 12 selisih nilai keluaran :

1. δ^t : selisih nilai dari keluaran pada unit LSTM yang sedang diproses
2. Δ^t : selisih nilai keluaran unit LSTM dan data asli
3. W_{hc}: nilai bobot keluaran pada aktivasi masukan unit LSTM
4. δz^{t+1}: selisih nilai masukan dari unit LSTM sebelumnya
5. W_{hi}: nilai bobot keluaran pada segmen memori pada unit LSTM
6. δi^{t+1}: selisih nilai segmen memori dari unit LSTM sebelumnya

7. W_{hf} : nilai bobot keluaran pada segmen forget pada unit LSTM
8. δf^{t+1}: selisih nilai segmen forget pada unit LSTM sebelumnya
9. W_{ho}: nilai bobot keluaran pada aktivasi keluaran pada unit LSTM
10. δo^{t+1}: selisih nilai pada aktivasi keluaran pada unit LSTM sebelumnya

Tahap selanjutnya adalah menghitung selisih nilai pada proses aktivasi *output gate* yang melibatkan pengaruhnya terhadap selisih keluaran unit saat ini dan turunan perhitungan nilai memori unit ini. Berikut persamaan 2 . 13 untuk menghitung selisih nilai aktivasi *output gate*.

$$\delta o^{-t} = \delta y^t . tanh(c^t) . \sigma(\bar{o}^t) \tag{2 . 13}$$

Berikut keterangan dari persamaan 2 . 13 selisih nilai aktivasi *output gate* :

1. $\delta \bar{o}^t$: selisih nilai dari aktivasi *output gate* pada unit LSTM yang sedang diproses
2. δy^t : selisih nilai dari keluaran pada unit LSTM
3. $tanh(c^t)$: nilai fungsi tanh terhadap nilai memori unit LSTM yang sedang diproses
4. $\sigma(\bar{o}^t)$: nilai fungsi turunan Sigmoid terhadap nilai penjumlahan keluaran pada unit LSTM yang sedang diproses

Tahap selanjutnya adalah menghitung nilai selisih memori unit LSTM yang sedang diproses saat ini mempertimbangkan pengaruhnya terhadap selisih keluaran unit ini, nilai aktivasi *output gate*, selisih nilai aktivasi memori, dan pengaruhnya terhadap perhitungan nilai memori unit berikutnya. Berikut persamaan 2 . 14 untuk menghitung selisih nilai memori.

$$\delta c^t = \delta y^t . o^t . tanh^{'}(c^t) + \delta c^{t+1} . f^{t+1} \tag{2 . 14}$$

Berikut keterangan dari persamaan 2 . 14 selisih nilai memori :

1. δc^t : selisih nilai dari memori unit LSTM yang sedang diproses
2. δy^t : selisih nilai dari keluaran pada unit LSTM yang sedang diproses

3. o^t: nilai *output gate* dari unit LSTM yang sedang diproses
4. $tanh^{'}(c^t)$: fungsi turunan *tanh* terhadap nilai memori unit LSTM yang sedang diproses
5. δc^{t+1} : selisih nilai dari memori pada unit LSTM sebelumnya
6. f^{t+1} = nilai dari *forget gate* pada unit LSTM sebelumnya

Tahap selanjutnya adalah menghitung selisih nilai aktivasi *forget gate* pada unit LSTM yang sedang diproses saat ini mempertimbangkan pengaruhnya terhadap nilai memori unit ini dan selisihnya. Berikut persamaan 2 . 15 untuk menghitung selisih nilai aktivasi *forget gate*.

$$\delta \bar{f}^t = \delta c^t . c^{t-1} . \sigma(\bar{f}^t) \quad (2 . 15)$$

Berikut keterangan dari persamaan 2 . 15 selisih nilai *forget gate* :

1. $\delta \bar{f}^t$: selisih nilai dari *forget gate* pada unit LSTM yang sedang diproses
2. δc^t : selisih nilai dari memori pada unit LSTM yang sedang diproses
3. $tanh(c^{t-1})$: nilai memori dari unit LSTM sebelumnya
4. $\sigma(\bar{f}^t)$: nilai turunan dari fungsi *sigmoid* terhadap nilai penjumlahan forget pada unit LSTM yang sedang diproses

Tahap selanjutnya adalah menghitung selisih nilai aktivasi *input gate* pada unit LSTM yang sedang diproses mempertimbangkan selisih nilai memori unit ini. Berikut persamaan 2 . 16 untuk menghitung selisih nilai aktivasi *input gate*

$$\delta \bar{i}^t = \delta c^t . z^t . \sigma(\bar{i}^t) \quad (2 . 16)$$

Berikut keterangan dari persamaan 2 . 16 selisih nilai aktivasi *input gate*:

1. $\delta \bar{i}^t$: selisih nilai dari *input gate* pada unit LSTM yang sedang diproses
2. δc^t : selisih nilai dari memori pada unit LSTM yang sedang diproses
3. z^t: nilai aktivasi masukan dari unit LSTM yang sedang diproses

4. $\sigma(\bar{i}^t)$: nilai turunan dari fungsi *Sigmoid* terhadap nilai penjumlahan *input gate* pada unit LSTM yang sedang diproses

Tahap terakhir dari mencari nilai selisih adalah menghitung selisih nilai aktivasi *memory gate* pada unit LSTM yang sedang diproses mempertimbangkan pengaruhnya terhadap selisih nilai memori unit ini. Berikut persamaan 2 . 17 untuk menghitung selisih nilai aktivasi *memory gate*.

$$\delta\bar{z}^t = \delta c^t . i^t . tanh^{'}(\bar{z}^t) \tag{2 . 17}$$

Berikut keterangan dari persamaan 2 . 17 selisih nilai *memory gate*:

1. $\delta\bar{z}^t$: selisih nilai aktivasi *memory gate* pada unit LSTM yang sedang diproses
2. δc^t : selisih nilai dari memori pada unit LSTM yang sedang diproses
3. i^t: nilai *input gate* pada unit LSTM yang sedang diproses
4. $tanh^{'}\bar{z}^t$: nilai turunan dari fungsi *tanh* terhadap nilai penjumlahan *memory gate* pada unit LSTM yang sedang diproses

Setelah didapatkan seluruh nilai selisih dari perhitungan unit LSTM, langkah berikutnya adalah menghitung nilai perubahan atau gradien yang akan digunakan untuk menentukan jumlah perubahan nilai pada bobot dan bias. Nilai ini didapatkan dengan menjumlahkan seluruh nilai selisih *gate* yang dikalikan terhadap nilai masukan dan keluaran dari unit sebelumnya. Berikut merupakan persamaan untuk mendapatkan perubahan nilai bobot dan bias yang digunakan dalam perhitungan LSTM.

$$\delta W_{x*} = \sum_{t=0}^{T} (\delta\bar{*}^t x^t) \tag{2 . 18}$$

$$\delta W_{h*} = \sum_{t=0}^{T-1} (\delta\bar{*}^t y^t) \tag{2 . 19}$$

$$\delta b_{*} = \sum_{t=0}^{T} (\delta\bar{*}^t) \tag{2 . 20}$$

Keterangan:

1. δW_{x*} : selisih nilai bobot dari masukan untuk perhitungan $*$ dimana $*$ merupakan substitusi dari simbol *gate* yang bersangkutan.
2. $\delta \bar{*}^t$: selisih nilai dari $*$ untuk unit LSTM yang sedang diproses
3. x^t : nilai masukan dari unit LSTM yang sedang diproses
4. δW_{h*} : selisih nilai bobot dari keluaran untuk perhitungan $*$
5. y^t : nilai keluaran dari unit LSTM yang sedang diproses
6. δb_* : selisih nilai bias untuk perhitungan $*$

Persamaan 2 . 18 merupakan persamaan untuk menghitung gradien nilai bobot yang digunakan untuk aktivasi masukan titik data unit LSTM yang sedang diproses dengan penjumlahan dari perkalian matriks nilai selisih terhadap titik data saat ini. Persamaan 2 . 19 merupakan persamaan untuk menghitung gradien nilai bobot yang digunakan untuk aktivasi masukan keluaran dari unit LSTM sebelumnya dengan penjumlahan dari perkalian matriks nilai selisih terhadap nilai keluaran dari unit LSTM sebelumnya. Persamaan 2 . 20 merupakan persamaan untuk menghitung gradien nilai bias yang digunakan dalam proses perhitungan aktivasi setiap *gate* yang didapat dengan menjumlahkan matriks nilai selisih untuk setiap *gate*.

Dengan demikian diperolehlah gradien untuk seluruh bobot dan bias yang akan diakumulasikan dengan nilai bobot dan bias sebelumnya untuk mendapatkan nilai bobot dan bias yang baru untuk iterasi selanjutnya.

2.1.8 Normalisasi Data

Algoritme dalam *machine learning* berusaha untuk menemukan kecenderungan data dengan membandingkan fitur-fitur pada data. Namun ketika fitur pada data berada pada skala yang berbeda maka akan terjadi masalah karena data yang lebih besar akan mendominasi data yang jauh lebih kecil. Karena itu diperlukan normalisasi data agar data yang akan diproses berada pada skala yang sama. Normalisasi bertujuan untuk menghilangkan redundansi dan menyeimbangkan fluktuasi data.

Z-score merupakan salah satu normalisasi data untuk menghindari data *outlier*. Salah satu kelebihan *Z-score* adalah untuk mengetahui probabilitas kemunculan skor yang terjadi dalam distribusi data normal. Normalisasi dengan *Z-score* akan menghasilkan data yang berada di rentang -3 sampai 3 [20].

Berikut persamaan normalisasi *Z-Score*.

$$ZScore = \frac{value - \mu}{\delta} \quad (2\ .\ 21)$$

Keterangan:

1. *value* = nilai data
2. μ = rata-rata data
3. δ = standar deviasi data

2.1.9 *Principal Component Analysis* (PCA)

PCA merupakan sebuah metode multivarians untuk menganalisis data yang dilakukan dengan beberapa variabel kuantitatif yang saling berkorelasi. Tujuan dari PCA adalah untuk mengekstraksi informasi penting dari data dan merepresentasikannya dalam bentuk variabel baru yang disebut dengan komponen utama. Komponen-komponen ini merupakan hasil representatif dari nilai korelasi antar fitur. Penggunaan PCA untuk mereduksi dimensi dapat dilakukan dengan menentukan *threshold* dari seberapa besar informasi fitur yang akan disimpan untuk digunakan. Nilai *threshold default* dari metode PCA adalah 95% [19]. Perhitungan secara lebih detail akan dijelaskan berikut dalam langkah-langkah perhitungan PCA.

Berikut langkah-langkah dalam analisis data menggunakan PCA.

1. Standarisasi data dengan fungsi normalisasi *Z-Score* menggunakan persamaan 2 . 21.
2. Menghitung matriks kovarians. Tujuan dari matriks kovarians adalah untuk menguji korelasi antar variabel dalam data. Matriks kovarians adalah matriks simetri p x p (p = jumlah dimensi) dengan entri berupa kombinasi dari data kovarians yang mungkin dari variabel awal. Berikut contoh matriks untuk data 3 dimensi dengan 3 variabel x, y, dan z.

$$\begin{pmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{pmatrix}$$

3. Menghitung *eigen vectors* dan *eigen values* dari matriks kovarians. Nilai *eigen values* dihitung dengan persamaan.

$$det(A - \lambda I) = 0 \quad (2.22)$$

Keterangan:

(a) λ = *eigen values*

(b) A = matriks data

(c) I = matriks identitas

Eigen vectors dihitung dengan persamaan.

$$det(A - \lambda I)x = 0 \quad (2.23)$$

Keterangan:

(a) x = *eigen vectors*

4. Data *eigen vector* diurutkan secara *descending* dari nilai *eigen values* yang terbesar. Setelah itu dipilih vektor fitur sesuai kebutuhan analisis. Pemilihan fitur dapat dilakukan dengan cara *explained variance percentage*. Persentase ini menghitung berapa banyak informasi (varians) dapat dihubungkan dengan komponen utama. Sebagai contoh berikut.

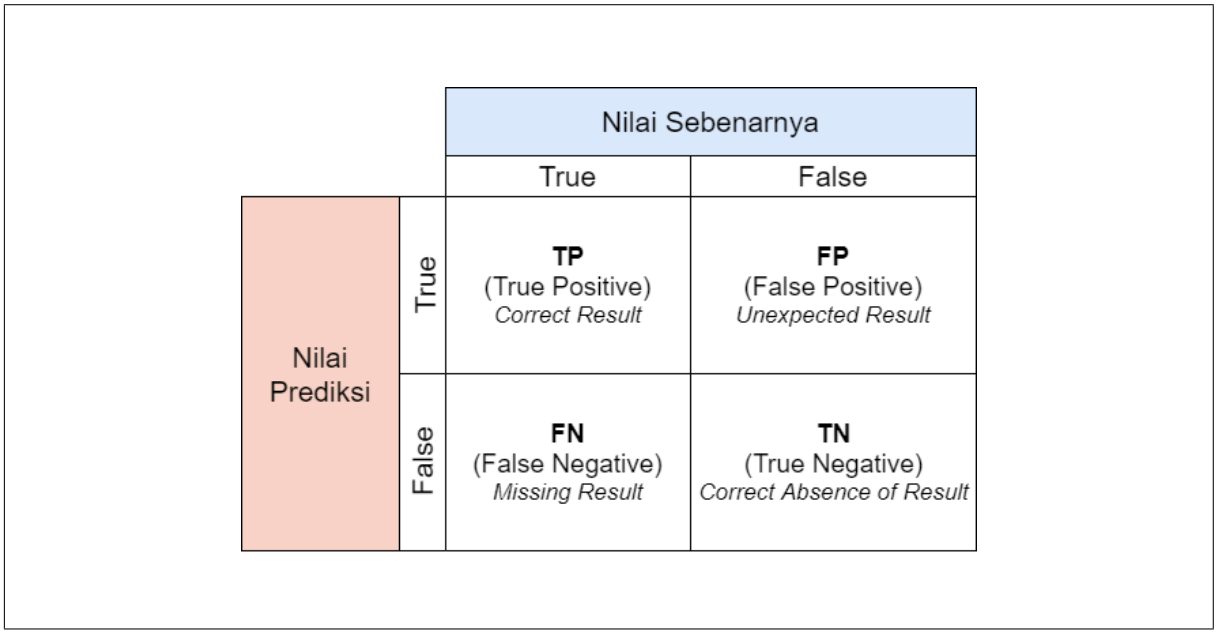
   Dataset awal memiliki 10 vektor fitur. Setelah menghitung matriks kovarians, diperoleh data :
   eigen values :[12, 10, 8, 7, 5, 1, 0.1, 0.03, 0.005, 0.0009].
   Jumlah total array ini adalah = 431359.
   5 nilai pertama mewakili: 42 / 431359 = 99.68% dari total data.

   Hal ini menunjukkan bahwa 5 *eigen vector* pertama memiliki porsi 99.68% dari varian atau informasi dari dataset. Data 5 vektor fitur terakhir dapat diabaikan atau dibuang karena hanya berisi 0.32% dari informasi.

2.1.10 Akurasi

Akurasi merupakan salah satu metode pengukuran statistik. Akurasi dalam statistika klasifikasi biner menyimpulkan seberapa baik sebuah sistem mengidentifikasi pengujian sesuai

dengan labelnya. Nilai akurasi dihitung dari jumlah nilai prediksi yang benar dibagi seluruh data.



Gambar 2.6 *Parameter Hasil Pengujian*

Berdasarkan 4 parameter hasil pengujian yang ditunjukkan oleh Gambar 2.6, nilai akurasi diperoleh dengan menjumlahkan nilai *true positive* dan *true negative* dan dibagi terhadap seluruh pengujian. Berikut adalah Persamaan 2 . 24 untuk menghitung nilai akurasi.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2 . 24}$$

2.1.11 *F-Measure*

F-Measure atau disebut juga $F_1 Score$ merupakan salah satu teknik pengukuran hasil pengujian dalam konsep statistika klasifikasi biner. Nilai dari *F-Measure* merupakan nilai *harmonic mean* dari *precision* dan *recall* yang ditunjukkan oleh Persamaan 2 . 27. *Precision* memiliki definisi tingkat ketepatan antara informasi yang diharapkan terhadap hasil yang diberikan yang ditunjukkan oleh Persamaan 2 . 25. *Recall* adalah tingkat keberhasilan sistem dalam menemukan informasi yang diharapkan dengan menggunakan Persamaan 2 . 26 [21].

Kelebihan dari *F-Measure* dalam menarik kesimpulan hasil pengujian adalah *F-Measure* mempertimbangkan *precision* dan *recall* sebagai satu kesatuan yang saling terhubung satu sama lain, sehingga fokus hasil pengujian dapat dipusatkan pada nilai *false positive*, *false negative*, dan *true positive*. Hal ini dapat menambahkan satu parameter untuk membuat analisis yang lebih mendalam terhadap hasil pengujian selain dari akurasi saja.

$$Precision = \frac{TP}{TP+FP} \quad (2.25)$$

$$Recall = \frac{TP}{TP+FN} \quad (2.26)$$

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (2.27)$$

2.2 *Library Apache Math 3*

Library yang digunakan dalam penelitian ini adalah *Library Apache Math 3* pada tahap *preprocessing*. *Library* ini digunakan untuk menghitung *eigen values* dan *eigen vector* untuk analisis *Principal Component Analysis* dalam tahap pemilihan fitur. Berikut merupakan fungsi yang digunakan dalam *Library Apache Math 3*.

Tabel 2.1 Tabel fungsi *Library Apache Math 3*

No	Fungsi	Deskripsi
1	Array2DRowRealMatrix (double[][] d);	Fungsi ini digunakan untuk membuat matriks baru menggunakan data masukan *array* sebagai data matriks dengan tipe RealMatrix. Parameter yang digunakan adalah *array* 2 dimensi yang merupakan nilai dari matriks yang ingin dibuat.
2	EigenDecomposition (RealMatrix matrix)	Fungsi ini digunakan untuk menghitung dekomposisi *eigen* dari *Real Matrix*. Parameter yang digunakan adalah RealMatrix dari matriks yang diproses.
3	getRealEigenvalues()	Fungsi ini digunakan untuk memperoleh nilai *eigen values* dari matriks asli.
4	getV()	Fungsi ini digunakan untuk memperoleh nilai matriks V. Kolom dalam matriks V merupakan *eigen vectors* dari matriks asli.

2.3 Tinjauan Studi

Pada Tabel 2.2 merupakan tabel *state-of-the-art* yang digunakan sebagai landasan studi dalam penelitian ini:

Tabel 2.2 Tinjauan Studi

No	Peneliti	Judul	Objektif	Hasil
1	Maniath, S., Ashok, A., Poornachandran, P., Sujadevi, V. G., Sankar, A. P., & Jan, S. [4]	Deep Learning LSTM Based Ransomware Detection (2017)	Menerapkan LSTM RNN untuk mendeteksi aktivitas *ransomware* dengan menganalisis panggilan API.	Peneliti dapat menerapkan LSTM untuk mendeteksi *ransomware* dengan *testing accuracy* 96.67%.
2	Pablo Torres, Carlos Catania, Sebastian Garcia, and Carlos Garcia Garino [3]	An Analysis of Recurrent Neural Network for Botnet Detection Behaviour(2016)	Menerapkan LSTM untuk mengatasi kekurangan RNN dalam mengatasi *imbalance network traffic* dan mencari panjang *state* koneksi yang optimal untuk mendeteksi *traffic behaviour*	Berhasil mendeteksi LSTM dengan *detection rate* untuk *window time* T=300s adalah 98.3% *true positive* terhadap *non-malicious* dan 99.9% *true positive* terhadap *malicious*.
3	Bontemps, L., McDermott, J., & Le-Khac, N. A. [14]	Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks (2016)	Menerapkan LSTM RNN untuk mendeteksi anomali dari beberapa aksi yang dilakukan sebelumnya oleh sebuah network traffic dalam waktu dan *threshold* tertentu.	Dengan *threshold* yang berbeda, dapat mencapai akurasi di antara 85-100% dengan tingkat *false alarm* yang rendah
4	David Zhao, Issa Traore, et al. [1]	Botnet Detection Based on Traffic Behavior Analysis and Flow Intervals (2013)	Menerapkan *decision tree classifier* untuk mendeteksi aktivitas *botnet* berdasrkan karakteristik dari *network flow* terutama interval waktu.	Berhasil menerapkan *decision tree classifier* dengan TPR di atas 90% dan FPR di bawah 5%.

No	Peneliti	Judul	Objektif	Hasil
5	Sherif Saad, Issa Traore, et al. [11]	Detecting P2P Botnets Through Network Behavior Analysis and Machine Learning (2014)	Menerapkan 5 teknik pembelajaran mesin untuk mendeteksi *malware botnet* pada fase *Command and Control*.	Berhasil menerapkan 5 teknik pembelajaran mesin untuk mendeteksi *malware botnet* dengan akurasi terendah di atas 88% dan *error rate* di bawah 20%.
6	Matija S and Jens M [12]	An Efficient Flow-Based Botnet Detection Using Supervised Machine Learning (2014)	Menerapkan 8 teknik pembelajaran mesin untuk mendeteksi *malware botnet* berdasarkan analisis *network flow*.	Berhasil mendeteksi *malware botnet* dengan akurasi 95.63% pada konfigurasi 1000 paket dan 60 detik durasi *flow* menggunakan *random tree classifier*.

Pada penelitian yang dilakukan Sumith Maniath et al., penelitian menerapkan RNN dengan LSTM untuk mendeteksi aktivitas *ransomware* dengan menganalisis panggilan API. *API Calls* diekstraksi melalui perekaman sebuah sistem yang sudah terinfeksi *ransomware* selama 56 jam dan menghasilkan 239 label *API Calls* yang berbeda. Kemudian data ini akan dijadikan masukan model LSTM dengan pembagian 80% data belajar dan 20% data uji. Konfigurasi model yang digunakan adalah 64 unit LSTM dengan 500 epoch. Hasil pengujian model terhadap data uji memperoleh akurasi 96.67%.

Pada penelitian yang dilakukan oleh Pablo Torres et al., penelitian menerapkan RNN dengan LSTM untuk mendeteksi *malware botnet*. Penelitian ini menganalisis perilaku atau sifat dari *malware botnet* berdasarkan beberapa fitur yaitu *Source IP Address*, *Destination IP Address*, *Destination Port*, *Protocol*, *Flow Size*, dan *Flow Duration*. Konfigurasi model LSTM yang digunakan adalah 128 unit LSTM, dengan *learning rate* 0.1 dan dilakukan sebanyak 30 *epoch*. Penelitian ini menghasilkan akurasi 98.3% dalam mengidentifikasi *non-malware* dan 99.9% dalam mengidentifikasi *malware* pada dataset CTU-13 dengan jumlah *malware* 188 dan jumlah *non-malware* 300 data.

Penelitian yang dilakukan Loic Bontemps et al., menerapkan LSTM RNN untuk mendeteksi anomali dari perekaman *network traffic* dalam waktu tertentu. Penelitian ini menggunakan dataset KDD 1999 yang merekam data *malware* dan *non-malware* selama 3 minggu untuk data belajar dan 2 minggu untuk data uji. Penelitian ini menggunakan konfigurasi 23 unit LSTM dengan learning rate 0.1. Akurasi penelitian ini dapat mencapai 85 sampai 100% dengan meningkatnya jumlah *false alarm* seiring bertambahnya jumlah *correct*

alarm.

Penelitian yang dilakukan David Zhao et al., menerapkan teknik pembelajaran mesin *decision tree classifier* untuk mendeteksi aktivitas *malware botnet* pada *network traffic* dengan menganalisis fitur-fitur dari *network flow* yang tercatat pada Lampiran B Tabel B-1. Penelitian ini diterapkan pada dataset yang berasal dari *Traffic Lab Ericson Research* dan *Lawrence Berkeley National Library*. Penelitian ini berhasil mencapai nilai TPR di atas 90% dengan beberapa konfigurasi berbeda dan FPR di bawah 5%.

Penelitian yang dilakukan Sherif Saad et al., menerapkan 5 teknik pembelajaran mesin sebagai perbandingan untuk mendeteksi *malware botnet*. 5 teknik ini mencakup *Nearest Neighbors Classifier* (NNC), *Linear Support Vector Machine* (SVM), *Artifical Neural Network* (ANN), *Gaussian Based Classifier* (GBC), dan *Naive Bayes Classifier* (NBC). Penelitian ini menganalisis *malware botnet* berdasarkan fitur dari *network flow* pada fase C&C yang dilampirkan pada Lampiran B Tabel B-2. Akurasi tertinggi diperoleh oleh SVM dengan akurasi di atas 97% dan *error rate* di bawah 6%.

Penelitian yang dilakukan Matija S dan Jens M, memperbandingkan 8 teknik pembelajaran mesin, yaitu *Naive Bayer Classifier* (NB), *Bayesian Network Classifier* (Bnet), *Logistic Regression* (LR), *Artificial Neural Networks* (ANN), *Support Vector Machine Linear Kernel* (SVMLin), *C4.5 Decision Tree* (C4.5), *Random Tree Classifier* (RTree), dan *Random Forest Classifier* (RForest) untuk mendeteksi malware *botnet*. Penelitian ini dilakukan terhadap dataset ISOP dengan sejumlah fitur yang terlampir pada Lampiran B Tabel B-3. Akurasi tertinggi diperoleh oleh RTree dengan akurasi 95.63% pada konfigurasi jumlah paket 1000 dan durasi flow 60 detik.

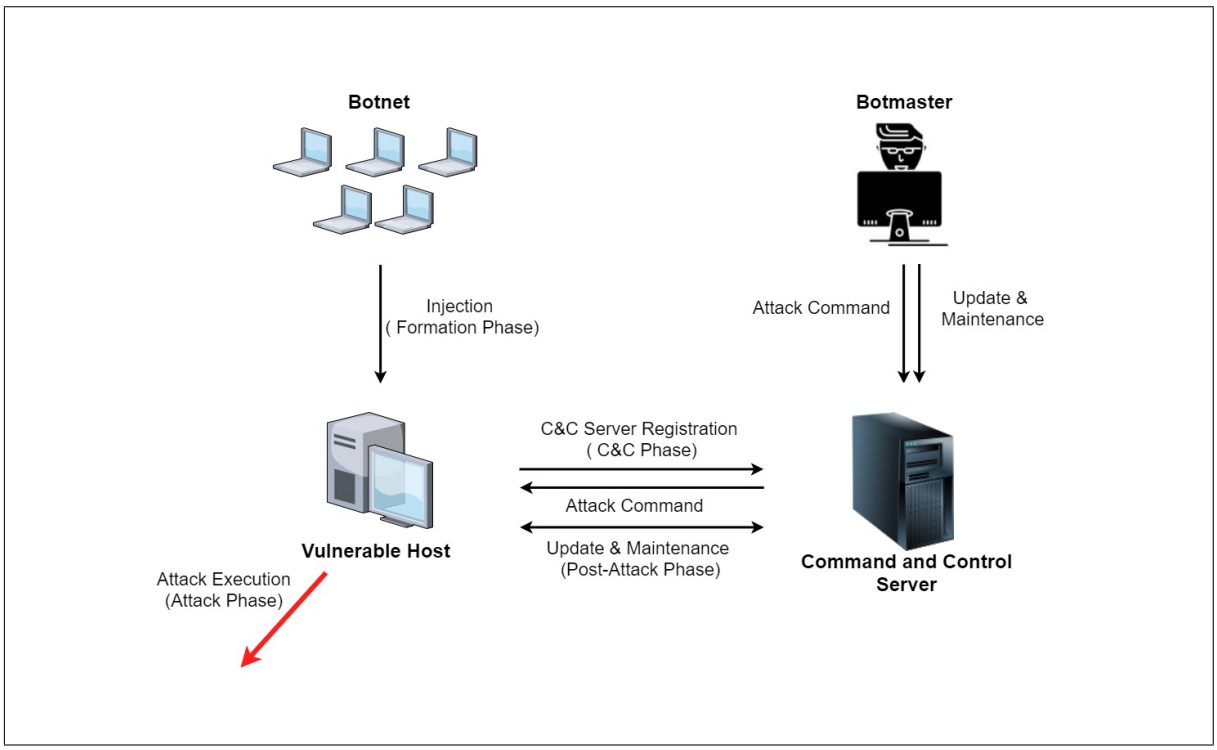
2.4 Tinjauan Objek

Pada bagian ini akan dijelaskan mengenai objek terkait yang akan digunakan dalam penelitian ini.

2.4.1 *Botnet*

Botnet adalah kumpulan dari komputer yang telah terinfeksi sehingga dapat dikontrol jarak jauh untuk mengeksekusi serangan yang terkoordinasi melalui sebuah *software malicious* yang disebut bot. Pengguna komputer yang terinfeksi jarang mengetahui adanya keberadaan dan aktivitas *malware botnet* dalam sistemnya.

Device atau komputer yang terinfeksi dikontrol oleh penyerang yang disebut *Botmaster*, dan biasanya digunakan untuk melaksanakan *cyberattack* seperti menjalankan *Distributed Denial of Service* (DDOS), mengirimkan *spam*, *click-fraud*, mencuri informasi pribadi, atau memanfaatkan komputer untuk menjalankan aksi yang terdistribusi [2].



Gambar 2.7 Fase hidup botnet

Botnet memiliki empat fase selama masa hidupnya, yaitu *formation*, *C&C*, *attack* dan *post-attack* [2]. Fase *formation* adalah fase awal, dimana penyerang memanfaatkan kerentanan suatu sistem untuk menginfeksi komputer korban dengan *script* yang menjalankan *malicious code* yang disebut *bot*. Selanjutnya *bot* akan mencoba untuk membentuk koneksi dengan *Command and Control server* dengan cara yang bervariasi untuk secara resmi bergabung sebagai bagian dari *botnet*. Fase *attack* atau penyerangan adalah sebuah fase dimana *bot* secara aktif melakukan *malicious attack* sesuai instruksi *botmaster*. Fase *post-attack* adalah fase terakhir dimana *botnet* diberi instruksi untuk melakukan pembaharuan terhadap *binary code*-nya untuk meningkatkan fungsionalitasnya dan mempertahankan diri dari sistem deteksi *malware*.

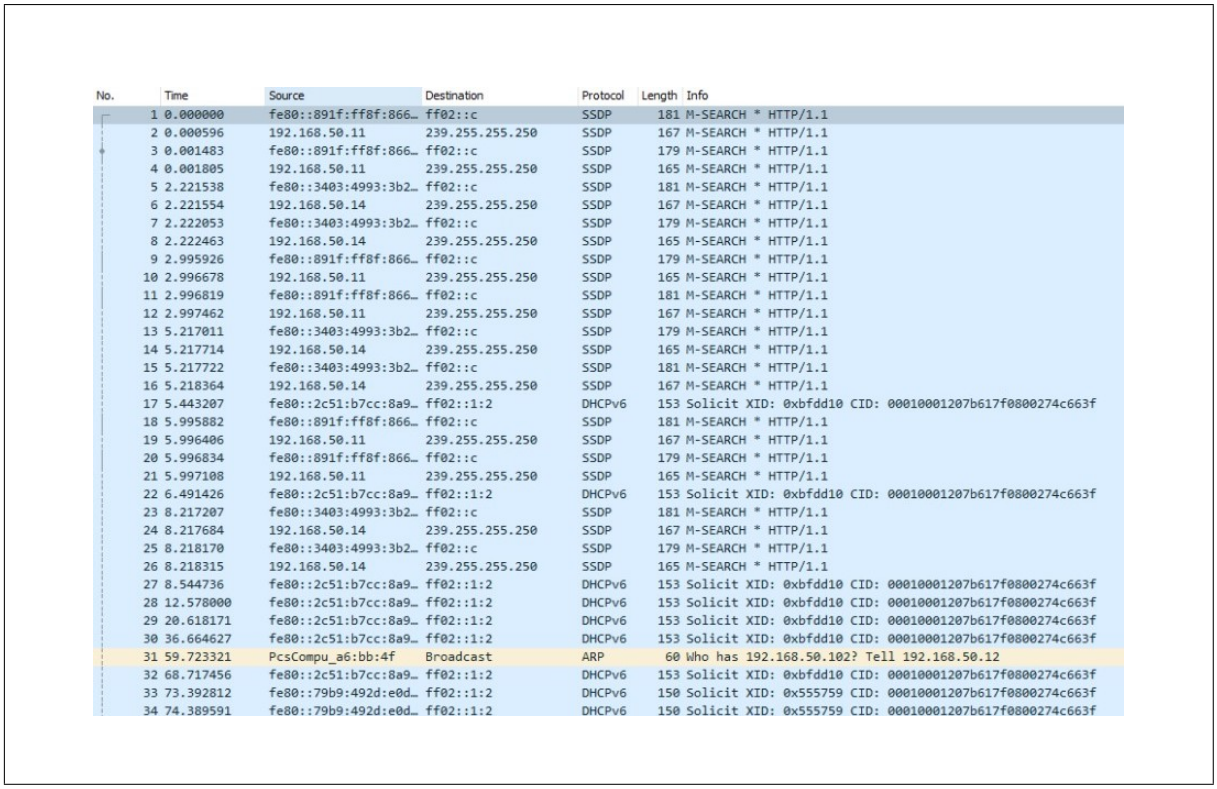
2.4.2 *Benign*

Benign merupakan tipe aplikasi normal yang tidak memiliki tujuan yang berbahaya atau tujuan yang merugikan pengguna aplikasi. Aplikasi *benign* hanya akan berjalan sesuai dengan dokumentasinya yang ditulis oleh *application developer* sebelumnya. Aplikasi *benign* juga pada umumnya dapat diunduh secara legal melalui *application store* terpercaya seperti Google PlayStore. Aplikasi *benign* telah melalui tahap pengecekan terlebih dahulu sebelum disebarkan ke penggunanya.

2.4.3 *Network Traffic*

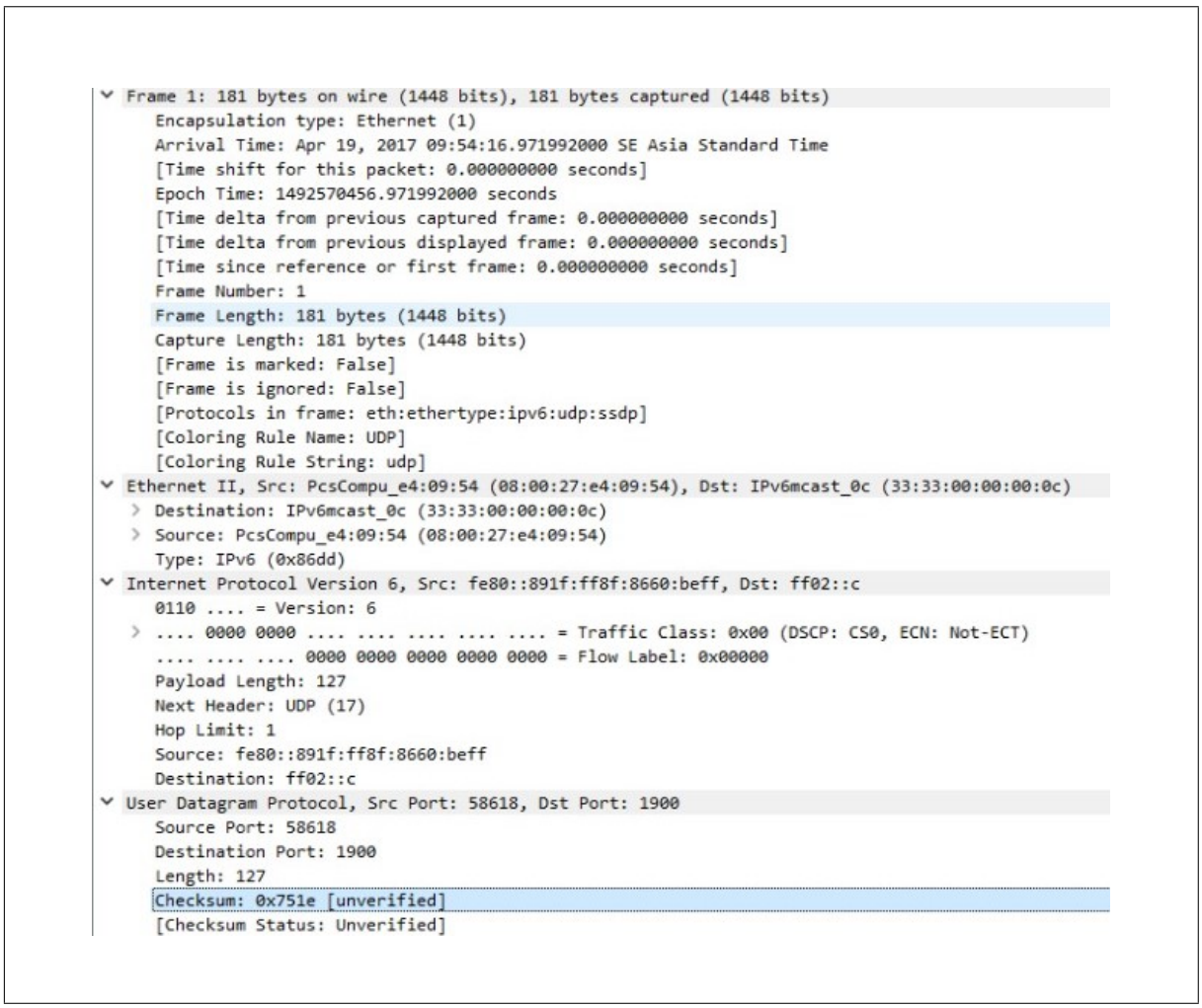
Network traffic adalah sejumlah data yang bergerak melintasi sebuah jaringan dalam kurun waktu tertentu. Umumnya, standar yang dipakai untuk berkomunikasi antar jaringan adalah *Transmission Control Protocol/Internet Protocol* (TCP/IP). *Network traffic* memiliki

fitur-fitur yang dapat digunakan untuk menganalisis lebih lanjut, dalam penelitian ini untuk mendeteksi adanya aktivitas botnet. Hasil perekaman *network traffic* disimpan dalam bentuk *file pcap (packet capture)*. Berikut merupakan contoh *Network Traffic* dalam *packet analyzer Wireshark*.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	fe80::891f:ff8f:866…	ff02::c	SSDP	181	M-SEARCH * HTTP/1.1
2	0.000596	192.168.50.11	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
3	0.001483	fe80::891f:ff8f:866…	ff02::c	SSDP	179	M-SEARCH * HTTP/1.1
4	0.001805	192.168.50.11	239.255.255.250	SSDP	165	M-SEARCH * HTTP/1.1
5	2.221538	fe80::3403:4993:3b2…	ff02::c	SSDP	181	M-SEARCH * HTTP/1.1
6	2.221554	192.168.50.14	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
7	2.222053	fe80::3403:4993:3b2…	ff02::c	SSDP	179	M-SEARCH * HTTP/1.1
8	2.222463	192.168.50.14	239.255.255.250	SSDP	165	M-SEARCH * HTTP/1.1
9	2.995926	fe80::891f:ff8f:866…	ff02::c	SSDP	179	M-SEARCH * HTTP/1.1
10	2.996678	192.168.50.11	239.255.255.250	SSDP	165	M-SEARCH * HTTP/1.1
11	2.996819	fe80::891f:ff8f:866…	ff02::c	SSDP	181	M-SEARCH * HTTP/1.1
12	2.997462	192.168.50.11	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
13	5.217011	fe80::3403:4993:3b2…	ff02::c	SSDP	179	M-SEARCH * HTTP/1.1
14	5.217714	192.168.50.14	239.255.255.250	SSDP	165	M-SEARCH * HTTP/1.1
15	5.217722	fe80::3403:4993:3b2…	ff02::c	SSDP	181	M-SEARCH * HTTP/1.1
16	5.218364	192.168.50.14	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
17	5.443207	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
18	5.995882	fe80::891f:ff8f:866…	ff02::c	SSDP	181	M-SEARCH * HTTP/1.1
19	5.996406	192.168.50.11	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
20	5.996834	fe80::891f:ff8f:866…	ff02::c	SSDP	179	M-SEARCH * HTTP/1.1
21	5.997108	192.168.50.11	239.255.255.250	SSDP	165	M-SEARCH * HTTP/1.1
22	6.491426	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
23	8.217207	fe80::3403:4993:3b2…	ff02::c	SSDP	181	M-SEARCH * HTTP/1.1
24	8.217684	192.168.50.14	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
25	8.218170	fe80::3403:4993:3b2…	ff02::c	SSDP	179	M-SEARCH * HTTP/1.1
26	8.218315	192.168.50.14	239.255.255.250	SSDP	165	M-SEARCH * HTTP/1.1
27	8.544736	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
28	12.578000	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
29	20.618171	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
30	36.664627	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
31	59.723321	PcsCompu_a6:bb:4f	Broadcast	ARP	60	Who has 192.168.50.102? Tell 192.168.50.12
32	68.717456	fe80::2c51:b7cc:8a9…	ff02::1:2	DHCPv6	153	Solicit XID: 0xbfdd10 CID: 00010001207b617f0800274c663f
33	73.392812	fe80::79b9:492d:e0d…	ff02::1:2	DHCPv6	150	Solicit XID: 0x555759 CID: 00010001207b617f0800274c663f
34	74.389591	fe80::79b9:492d:e0d…	ff02::1:2	DHCPv6	150	Solicit XID: 0x555759 CID: 00010001207b617f0800274c663f

Gambar 2.8 Contoh paket *Network Traffic* pada aplikasi *Wireshark*



Gambar 2.9 Contoh detail paket *Network Traffic* pada aplikasi *wireshark*

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Masalah

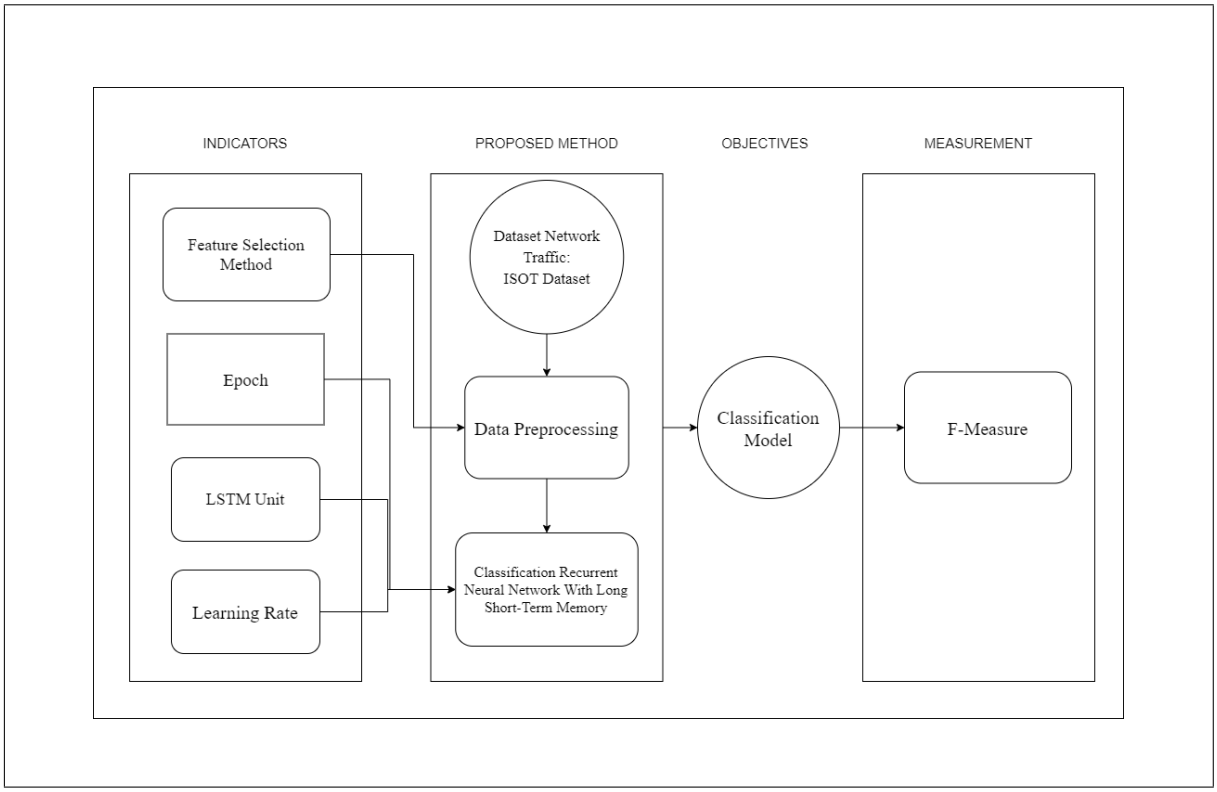
Seiring berkembangnya persebaran *malware botnet* dalam jaringan internet, diperlukan juga perkembangan sistem yang mampu mendeteksi *botnet* dalam sebuah jaringan. *Botnet* pada umumnya memiliki keseragaman perilaku dalam sebuah jaringan, yang mengacu pada siklus masa hidup dari *botnet* itu sendiri. Dengan pola yang dimiliki oleh *botnet* ini, peneliti berasumsi bahwa metode *deep learning* RNN dengan LSTM mampu dengan mendeteksi aktivitas *malware botnet* dalam sebuah jaringan dengan mempelajari perilakunya dalam fitur-fitur dari *network traffic*. Penelitian ini akan membangun sebuah sistem klasifikasi *malware botnet* yang akan menerima masukan data berupa hasil perekaman *network traffic* dalam kurun waktu tertentu. Data tersebut mengandung seluruh informasi *network traffic* yang berlangsung dan direkam dalam 84 fitur yang berbeda. Pendekatan yang digunakan untuk melakukan penelitian ini adalah pendekatan pembelajaran mesin dengan metode LSTM. Keluaran yang dihasilkan dari penelitian ini adalah klasifikasi yang terbagi menjadi 2 kelas yaitu *botnet* dan *benign*.

Tahap awal dari sistem ini adalah melakukan *preprocessing* terhadap data yang akan dilatih maupun diuji. Dengan adanya tahap *preprocessing*, data akan disesuaikan dengan kebutuhan dan hanya memiliki bagian penting yang diperlukan oleh sistem. *Prepocessing* yang dilakukan adalah *feature selection* dan normalisasi data. *Feature selection* akan dilakukan menggunakan 4 pendekatan yaitu pemilihan berdasarkan literatur, pemahaman peneliti, gabungan dari literatur dan pemahaman peneliti, serta menggunakan metode PCA. Kemudian tahap normalisasi data akan menggunakan fungsi normalisasi *z-score* untuk mengubah distribusi nilai data dengan rentang antara -3 dan 3. Tahap *preprocessing* akan menghasilkan data latih dan data uji.

Setelah tahap *preprocessing*, tahap berikutnya adalah tahap pelatihan. Pada tahap ini, data latih akan dijadikan sebagai masukan pada model LSTM. Model LSTM memiliki bobot dan bias yang akan diinisialisasi secara acak dengan mempertimbangkan nilai data. Proses pelatihan akan dilakukan secara berulang sejumlah nilai *epoch* dengan konfigurasi jumlah *hidden layer* dan *learning rate* yang akan ditetapkan sebelumnya. Hasil dari pelatihan ini adalah model LSTM berupa matriks bobot serta bias yang akan digunakan dalam proses pengujian. Tahap terakhir adalah pengujian. Pengujian dilakukan dengan memasukkan data uji ke dalam model LSTM dengan bobot hasil tahap pelatihan. Hasil keluaran model LSTM kemudian akan diklasifikasi dan diberi label yang bertujuan untuk menyimpulkan jenis aktivitas, baik *botnet* maupun *benign*.

3.2 Kerangka Pemikiran

Berikut merupakan kerangka pemikiran dalam penelitian ini.

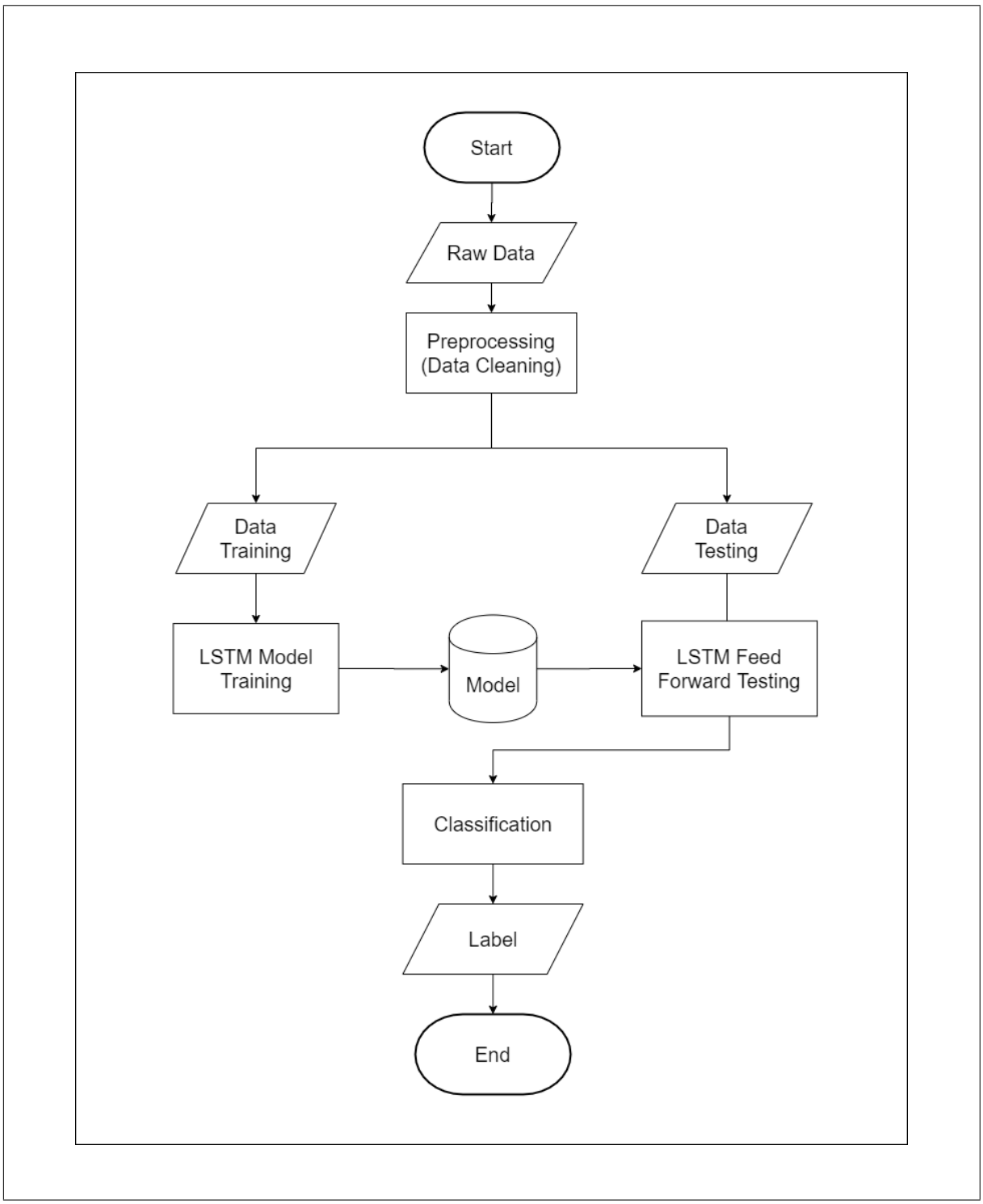


Gambar 3.1 Kerangka Pemikiran

Gambar 3.1 memaparkan bagian-bagian dari kerangka pemikiran yaitu indikator, *proposed method*, *objective*, dan *measurement*. Indikator menjelaskan faktor yang mempengaruhi hasil pada bagian objektif, di antaranya *Feature Selection Method* yang akan digunakan untuk tahap *preprocessing* data dalam menentukan metode pemilihan fitur serta jumlah fitur yang akan digunakan, dan parameter yang memiliki pengaruh pada pembangunan *neural network* seperti jumlah *epoch*, *learning rate* dan jumlah unit LSTM. Pada *proposed method*, terdapat dataset ISOT berupa file pcap (*packet capture*) yang berisi aktivitas dari *network traffic* botnet dan HTTP *application*. File pcap akan diolah dan dikonversi menjadi file .csv (*comma separated value*). Kemudian fitur-fitur pada network traffic akan dianalisis dan dipilih untuk dilanjutkan ke tahap pelatihan dan pengujian oleh model LSTM. Hasil pengujian akan dianalisis dengan pengukuran *F-measure*, yang menghasilkan nilai *true positive*, *true negative*, *false negative*, *false positive*, *recall*, *precision*, akurasi, dan *F-score*. Dalam penelitian ini nilai positif mengacu kepada *botnet* dan negatif mengacu kepada *benign*.

3.3 Urutan Proses Global

Berikut merupakan *flowchart* yang menggambarkan urutan proses global dalam penelitian ini.



Gambar 3.2 *Flowchart* Global

Berikut urutan proses global sesuai dengan *Flowchart* Global pada Gambar 3.2.

1. Langkah pertama, *raw dataset* yang digunakan akan melalui tahap *preprocessing* untuk memperoleh data yang siap digunakan sebagai masukan perhitungan *neural network* LSTM. Data akan melalui tahap pemilihan fitur dan normalisasi menghasilkan data latih dan data uji.
2. Langkah berikutnya, data latih akan menjadi masukan untuk proses pelatihan model LSTM

untuk memperoleh nilai bobot yang optimal. Hasil tahap ini adalah sebuah berkas model LSTM dengan bobot dan bias yang sudah diperbaharui dan siap untuk diuji.

3. Kemudian, data uji akan menjadi masukan untuk proses pengujian LSTM dengan menggunakan model LSTM dari proses pelatihan. Keluaran dari proses ini adalah keluaran dari hasil perhitungan LSTM untuk kemudian dilakukan proses klasifikasi.
4. Tahapan terakhir adalah menggunakan hasil keluaran tahap pengujian untuk proses klasifikasi. Hasil tahapan klasifikasi adalah label untuk masing-masing keluaran titik data yang terbagi antara label *botnet* dan *benign*.

3.4 Data Sampel

Data yang digunakan dalam penelitian ini berasal dari ISOT Dataset. *Dataset* ISOT diperoleh dari *ISOT Research Lab, University of Victoria, Canada*. *Dataset* ini direkam dari sebuah *virtual machine* yang dijalankan selama beberapa hari dimana tercakup di dalamnya 9 jenis *botnet* yang berbeda dan aplikasi-aplikasi yang umum digunakan. Dataset ISOT terdiri dari 87854 *flow* dengan total 84 fitur yang terlampir pada Lampiran A Tabel A-1.

Berikut merupakan contoh data dari beberapa fitur pada *dataset* ISOT.

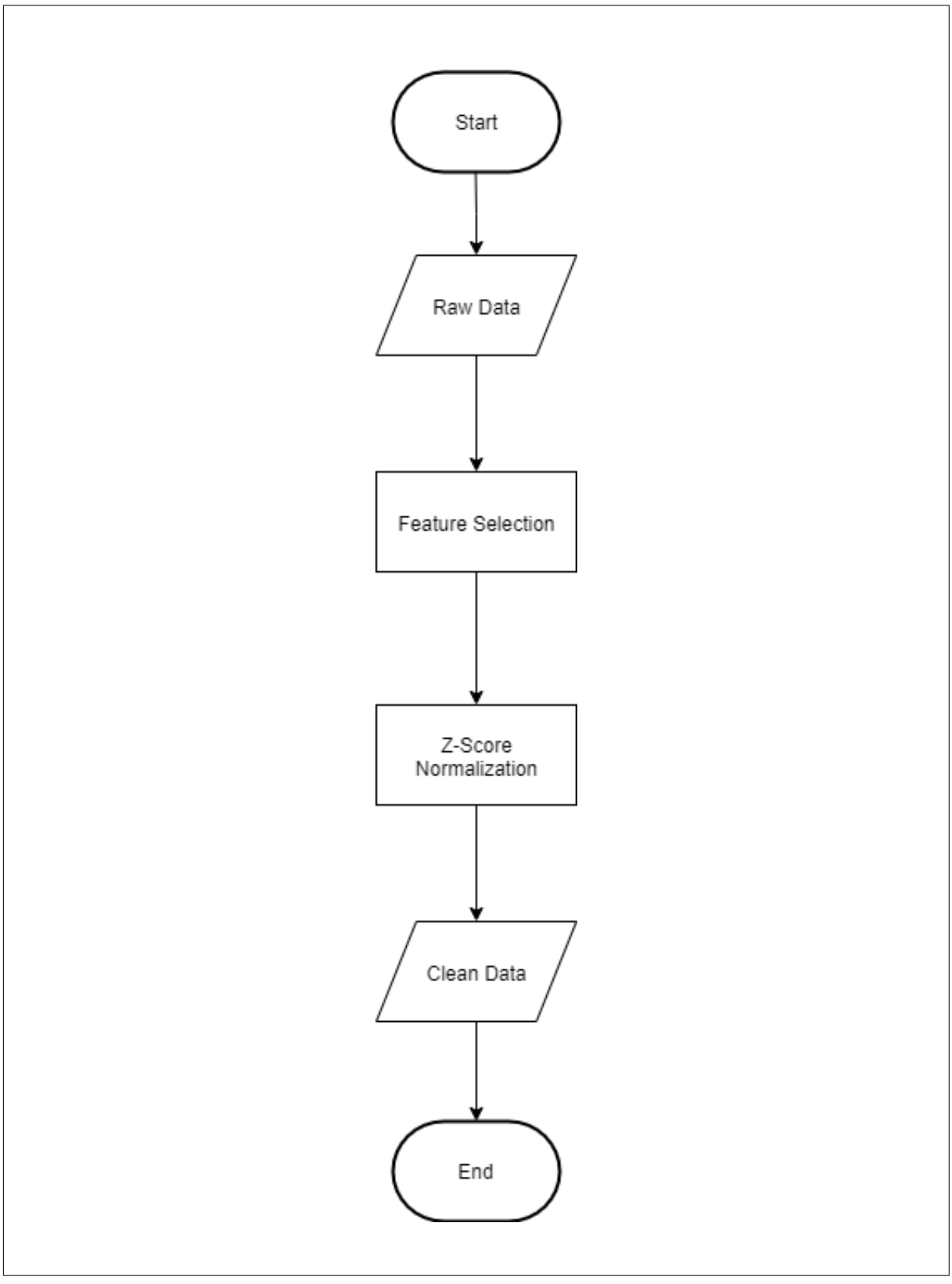
Tabel 3.1 Contoh Data Network Traffic

Flow ID	Source IP	Source Port	Destination IP	Destination Port	Flow Byte/s
192.168.50.88-8.8.8.8-34874-53-6	1921685088	34874	8888	53	4.62011E+15
192.168.50.88-8.8.8.8-56291-53-6	1921685088	56291	8888	53	7.34151E+15
192.168.50.88-8.8.8.8-43734-53-6	1921685088	43734	8888	53	7.11821E+15
192.168.50.88-8.8.4.4-41224-53-6	1921685088	41224	8844	53	7.13645E+15
192.168.50.88-8.8.4.4-52585-53-6	1921685088	52585	8844	53	7.28485E+15

Bentuk data yang terdapat pada *dataset* ini pada umumnya bersifat kurang stabil, karena nilai dapat meningkat dan menurun dengan sangat tajam sesuai dengan karakteristik *network traffic* pada umumnya yang fluktuatif bergantung erat dengan aktivitas yang akan dilakukan. Oleh karena itu masih diperlukan pemrosesan sebelum data ini dapat digunakan secara optimal untuk pembelajaran mesin.

3.5 *Preprocessing* Data

Tahap *preprocessing* dalam penelitian ini dilakukan untuk mengolah *dataset* agar siap dijadikan masukan bagi model LSTM. Berikut merupakan *flowchart preprocessing* data.



Gambar 3.3 *Flowchart Preprocessing*

Gambar 3.3 menunjukkan proses data yang diolah dengan melalui tahap pemilihan fitur dan *Z-Score Normalization*. Proses *preprocessing* dimulai dengan pemilihan fitur yang terbagi menjadi lima analisis yaitu :

1. Pemilihan Fitur Irisan Literatur
2. Pemilihan Fitur oleh Peneliti
3. Penggabungan Fitur Irisan Literatur dan Peneliti
4. Pemilihan Fitur Gabungan Literatur
5. Reduksi Dimensi PCA

Lima analisis ini akan diterapkan dengan tujuan untuk membandingkan pendekatan pemilihan fitur-fitur dataset yang menghasilkan akurasi terbaik. Pada bagian selanjutnya akan dijelaskan proses pemilihan fitur dan normalisasi data secara lebih rinci.

3.5.1 Pemilihan Fitur

Tahap pertama dari *preprocessing* pada penelitian ini adalah seleksi fitur untuk mereduksi fitur-fitur yang kurang optimal untuk digunakan sebagai masukan model LSTM. Fitur yang digunakan untuk pembelajaran dan pengujian oleh model LSTM akan dipilih berdasarkan lima analisis yang sudah disebutkan sebelumnya. Berikut akan dijelaskan metode-metode pemilihan fitur secara detail beserta alasannya.

3.5.1.1 Fitur Irisan Literatur

Pemilihan fitur berdasarkan irisan literatur, dalam penelitian ini akan disebut sebagai fitur irisan literatur, merupakan sejumlah fitur yang digunakan bersama oleh beberapa penelitian sebelumnya dalam mendeteksi *botnet* berdasarkan fitur-fitur dari *network traffic*. Analisis ini digunakan bertujuan untuk mencapai hasil pengujian terbaik mempertimbangkan fitur yang dapat merepresentasikan aktivitas *malware botnet* yang digunakan penelitian-penelitian yang pernah dilakukan terhadap fitur dari *network traffic*. Untuk mendapatkan set fitur irisan literatur, maka akan dipilih fitur-fitur irisan dari beberapa referensi penelitian terkait deteksi *malware* botnet melalui fitur *network traffic* yang tercantum pada Lampiran B. Terdapat 9 fitur hasil irisan yang akan digunakan pada penelitian ini sebagai fitur pertama dan dinamakan sebagai fitur literatur. Fitur-fitur ini adalah SrcPort, Packet Length Mean, Total FWD packets, Total BWD Packets, Flow packets/s, Flow byte/s, Flow Duration, Flow IAT Mean, dan Flow IAT STD. Berikut merupakan daftar 9 fitur irisan literatur beserta deskripsinya.

Tabel 3.2 Sembilan Fitur Literatur

No	Nama Fitur	Deskripsi Fitur	Referensi
1	SrcPort	Port pengirim	[11], [1], [12]
2	Packet Length Mean	Rata-rata ukuran packet dalam satu detik	[11], [1]
3	Total FWD Packets	Jumlah paket yang dikirim	[11], [1]
4	Total BWD Packets	Jumlah paket yang diterima	[11], [1]
5	Flow Packets/s	Jumlah pertukaran paket setiap detik dalam satu flow	[11], [1]
6	Flow Byte/s	Jumlah bytes setiap detik	[11], [1]
7	Flow Duration	Durasi flow	[1], [12]
8	Flow IAT Mean	Rata-rata waktu dari satu paket ke paket lain	[1], [12]
9	Flow IAT Std	Standar deviasi antar paket	[1], [12]

3.5.1.2 Fitur Peneliti

Fitur yang akan diajukan berikutnya, dalam penelitian ini selanjutnya akan disebut fitur peneliti, didasarkan pada pengetahuan dan pengertian peneliti mengenai 84 fitur *network traffic* yang tercantum pada Lampiran A Tabel A-1 dan karakteristik aktivitas *botnet* serta hubungan dari keduanya yang dapat menandai adanya aktivitas *botnet* dalam sebuah *flow network traffic*. Fitur yang akan dianalisis oleh peneliti untuk menjadi fitur peneliti berjumlah 84 fitur dikurangi dengan 9 fitur yang sudah digunakan oleh fitur literatur, sehingga terdapat 75 fitur yang akan dianalisis oleh peneliti. Peneliti kemudian memilih 8 fitur yang diasumsikan memiliki ciri-ciri yang menandakan adanya aktivitas *malware botnet* dalam *flow network traffic*.

Lima fitur pertama berkaitan dengan paket yang diterima dan dikirim oleh *network flow* yaitu FWD Packet Length Max, Total Length of BWD Packets, FWD Packet/s, BWD Packet/s, dan Packet Length Variance. FWD Packet Length Max merupakan panjang maksimal dari sebuah paket yang dikirim dalam bytes, fitur ini dapat mengindikasikan adanya pola dari *botnet* dalam mengirimkan ukuran paket yang tidak biasa. Total Length of BWD Packets merupakan jumlah total bytes dari seluruh paket yang diterima, botnet yang terhubung pada C&C *server* akan memiliki kesamaan total *bytes* anomali yang diterima. FWD Packet/s dan BWD Packet/s merupakan jumlah paket yang dikirim dan diterima setiap detik, fitur ini dapat berkaitan dengan aktivitas dari *bot client* dan C&C server yang memiliki pola pengiriman dan penerimaan paket yang serupa. Packet Length Variance adalah nilai varians atau persebaran ukuran paket dalam *flow network traffic*, *bot client* akan memiliki persebaran ukuran fitur yang berbeda dengan *non-malicious client* disebabkan oleh adanya pengiriman paket yang tidak biasa.

Tiga Fitur terakhir berkaitan dengan waktu, baik waktu pertukaran paket maupun durasi aktif dan *idle* dari *network flow*. *Active mean* merupakan waktu rata-rata *flow* aktif sampai kondisi idle dan *Idle mean* merupakan waktu rata-rata *flow idle* sampai aktif kembali.

Kedua fitur ini dapat menjadi indikator adanya aktivitas botnet karena *bot client* pada satu jaringan *botnet* yang sama akan memiliki kemiripan waktu aktif dan waktu *idle* saat menunggu perintah dari C&C *server*. FWD IAT Std merupakan standar deviasi jarak waktu pertukaran satu paket ke paket lainnya. *Bot client* akan menerima automasi perintah dari *Bot Master* melalui C&C*server*, sehingga menghasilkan jarak waktu pengiriman paket yang memiliki pola tertentu. 8 fitur inilah yang dipilih oleh peneliti untuk dijadikan masukan bagi proses pembelajaran mesin LSTM, yang diasumsikan dapat meningkatkan kualitas pembelajaran terhadap aktivitas *botnet* dan *benign*. Berikut merupakan tabel hasil pemilihan fitur peneliti beserta deskripsinya.

Tabel 3.3 Fitur Pilihan

No	Nama Fitur	Deskripsi Fitur
1	FWD Packet Length Max	Panjang maksimal sebuah paket yang dikirim dalam bytes
2	Total Length of BWD Packets	Jumlah total bytes dari seluruh paket yang diterima
3	FWD Packet/s	Jumlah paket yang dikirim setiap detik
4	BWD Packet/s	Jumlah paket yang diterima setiap detik
5	Packet Length Variance	Nilai varians dari ukuran paket
6	Active Mean	Waktu rata-rata aktif sampai kondisi idle
7	Idle Mean	Waktu rata-rata kondisi idle sampai aktif
8	FWD IAT Std	Standar deviasi Inter-Arrival Time/jarak waktu pengiriman antar paket

3.5.1.3 Fitur Kombinasi

Fitur berikut ini, yang selanjutnya dalam penelitian ini akan disebut fitur kombinasi, merupakan hasil penggabungan fitur irisan literatur dan fitur pilihan peneliti, sehingga berjumlah sebanyak 17 fitur. Fitur ini diharapkan dapat memberikan hasil pembelajaran yang lebih akurat terhadap aktivitas *botnet* dalam *flow network traffic*. Berikut merupakan daftar fitur kombinasi beserta deskripsinya.

Tabel 3.4 Fitur Kombinasi

No	Nama Fitur	Deskripsi Fitur
1	SrcPort	Port pengirim
2	Packet Length Mean	Rata-rata ukuran packet dalam satu detik
3	Total FWD Packets	Jumlah paket yang dikirim

4	Total BWD Packets	Jumlah paket yang diterima
5	Flow Packets/s	Jumlah pertukaran paket setiap detik dalam satu flow
6	Flow Byte/s	Jumlah bytes setiap detik
7	Flow Duration	Durasi flow
8	Flow IAT Mean	Rata-rata waktu dari satu paket ke paket lain
9	Flow IAT Std	Standar deviasi antar paket
10	FWD Packet Length Max	Panjang maksimal sebuah paket yang dikirim dalam bytes
11	Total Length of BWD Packets	Jumlah total bytes dari seluruh paket yang diterima
12	FWD Packet/s	Jumlah paket yang dikirim setiap detik
13	BWD Packet/s	Jumlah paket yang diterima setiap detik
14	Packet Length Variance	Variasi ukuran dari setiap paket
15	Active Mean	Waktu rata-rata aktif sampai kondisi idle
16	Idle Mean	Waktu rata-rata kondisi idle sampai aktif
17	FWD IAT Std	Standar deviasi Inter-Arrival Time/jarak waktu pengiriman antar paket

3.5.1.4 Fitur Gabungan Literatur

Fitur Gabungan Literatur merupakan fitur hasil penggabungan fitur-fitur yang digunakan oleh penelitian-penelitian terkait dengan deteksi *malware botnet* berdasarkan fitur *network traffic*. Fitur ini merupakan gabungan dari 3 set fitur yang tercantum pada lampiran B, yang sebelumnya digunakan sebagai acuan dalam menentukan fitur irisan literatur. Analisis pemilihan fitur ini digunakan untuk mencoba hasil pengujian terhadap analisis pemlihan fitur yang berbeda untuk mencapai hasil yang terbaik berdasarkan fitur-fitur yang telah dianalisis oleh penelitian sebelumnya. Fitur gabungan literatur ini berjumlah 27 fitur yang akan dijelaskan dalam Tabel 3.5 sebagai berikut.

Tabel 3.5 Fitur Kombinasi

No	**Nama Fitur**	**Deskripsi Fitur**
1	SrcIP	Alamat IP pengirim

2	SrcPort	Port pengirim
3	Destination IP	Alamat IP yang dituju
4	Destination Port	Port yang dituju
5	Protocol	Nomor protokol layer transport
6	Flow Duration	Total durasi flow
7	Packet Length Mean	Rata-rata panjang paket dalam flow
8	Total FWD Packets	Jumlah paket yang dikirim
9	Total BWD Packets	Jumlah paket yang diterima
10	Packet Length Variance	Nilai varians panjang paket dalam flow
11	Down/Up Ratio	Rasio upload and download
12	Flow Packets/s	Jumlah paket per detik dalam satu flow
13	Flow Bytes/s	Jumlah byte per detik dalam satu flow
14	Min Packet Length	Panjang minimal paket dalam flow
15	Flow IAT Mean	Nilai rata-rata jarak waktu antar paket
16	Flow IAT Std	Standar deviasi jarak waktu antar paket
17	Subflow FWD Packets	Rata-rata jumlah paket dalam setiap subflow yang dikirim
18	Average Packet Size	Rata-rata ukuran paket
19	FIN Flag Count	Jumlah Finish Flag
20	ECE Flag Count	Jumlah ECE TCP Flag
21	Total Length of FWD Packets	Total panjang paket yang dikirim
22	FWD Packet Length Mean	Rata-rata panjang paket yang dikirim
23	FWD Packet Length Min	Nilai minimal panjang paket yang dikirim
24	FWD Packet Length Std	Standar deviasi panjang paket yang dikirim

25	SYN Flag Count	Jumlah Synchronize Flag
26	ACK Flag Count	Jumlah Acknowledged Flag
27	Flow IAT Min	Nilai minimal jarak waktu antar paket

3.5.1.5 Fitur PCA

Fitur PCA merupakan fitur pilihan hasil analisis menggunakan metode PCA. Metode ini digunakan oleh peneliti karena kemampuannya untuk mereduksi dimensi fitur dengan mempertahankan informasi-informasi penting melalui korelasi setiap fitur. Fitur hasil reduksi dimensi dengan metode PCA akan menghasilkan sejumlah fitur baru sesuai *threshold* yang ditetapkan oleh peneliti. Dalam penelitian ini, nilai *threshold* yang digunakan adalah nilai *default* yang telah dijelaskan pada Bab 2.1.9 yaitu 95%. Berikut merupakan *pseudocode* dari implementasi fitur PCA.

Algoritme 1 Proses ektraksi fitur dengan metode PCA

1. Masukan berupa matriks data yang sudah dinormalisasi dengan persamaan 2 . 21.
2. Menghitung matriks kovarians dengan menghitung nilai kovarian antara 2 entri data untuk setiap nilai masukan.
3. Menghitung nilai *eigen vector* dengan persamaan 2 . 23 dan *eigen values* dengan persamaan 2 . 22 terhadap matriks kovarian.
4. Mengurutkan *eigen vector* berdasarkan nilai *eigen values* yang terbesar.
5. Mereduksi dimensi dengan menjumlahkan *eigen values* sampai mencapai nilai *threshold* yang diinginkan dan hanya menyimpan nilai *eigen vector* yang bersangkutan.
6. Melakukan perkalian antara *eigen vector* yang sudah direduksi dengan matriks data masukan untuk menghasilkan data baru.
7. Menyimpan matriks data baru sebagai set data.

Berikut merupakan contoh fitur hasil dari analisis dengan metode PCA.

-0.02507	-0.62462	0.1156	-0.55855	0.476089	-0.14328	-0.17769	-0.3625	0.055113	0.192971	0.027283
-0.03464	-0.76624	0.144708	-0.56714	0.559647	-0.14257	-0.23081	-0.43992	0.059305	0.206423	-0.00273
-0.03678	-0.79564	0.127422	-0.46145	0.599353	-0.24174	-0.2178	-0.401	0.051051	0.247452	-0.07992
-0.03802	-0.82322	0.128559	-0.46675	0.650372	-0.32602	-0.20364	-0.41477	0.041123	0.343337	-0.3143
-0.02479	-0.63451	0.121031	-0.60138	0.477762	-0.13693	-0.18024	-0.37709	0.056624	0.21954	-0.06967
-0.02393	-0.629	0.118976	-0.61466	0.477039	-0.12859	-0.1921	-0.37671	0.055272	0.199726	0.018588
-0.03022	-0.69282	0.122791	-0.51754	0.548061	-0.23863	-0.14807	-0.39014	0.050088	0.297395	-0.254
-0.03635	-0.81663	0.13982	-0.52979	0.658791	-0.30251	-0.13449	-0.4714	0.030381	0.391654	-0.41244
-0.03495	-0.80058	0.141785	-0.56386	0.565239	-0.1269	-0.21371	-0.4471	0.05684	0.201947	0.025576
-0.00471	-0.75149	0.055123	-0.54644	0.519101	-0.07241	-0.06559	-0.46412	0.035807	0.218898	-0.1109
-0.03508	-0.80226	0.142409	-0.57293	0.570053	-0.14943	-0.2122	-0.44452	0.058914	0.234095	-0.08485
-0.03594	-0.83088	0.146173	-0.58836	0.58945	-0.13828	-0.22222	-0.46655	0.054519	0.225335	-0.0192
-0.02028	-0.65687	0.132927	-0.83102	0.487627	-0.06724	-0.27197	-0.43356	0.057712	0.205723	0.054485
-0.03826	-0.80462	0.1286	-0.43022	0.625256	-0.30882	-0.17314	-0.40346	0.044022	0.332456	-0.33392
0.032905	-0.19663	0.060195	-1.42041	0.127381	0.306721	-0.36547	-0.40434	0.067512	0.173593	-0.22615
0.04197	0.586295	-0.12735	0.101841	-0.10077	-0.31725	0.138968	0.325991	-0.11873	0.28622	-0.2282
0.084493	0.656635	-0.10636	-1.28488	-0.03888	-0.08441	-0.15205	0.144932	0.035649	0.515169	-0.15346
-0.03103	-0.72889	0.134201	-0.56708	0.514163	-0.11357	-0.18754	-0.41386	0.059241	0.198789	-0.01521
-0.03599	-0.78536	0.133549	-0.4883	0.552481	-0.14866	-0.21848	-0.40754	0.06077	0.180235	0.053509

Gambar 3.4 Contoh Fitur PCA

3.5.2 Normalisasi Data

Tahap selanjutnya adalah melakukan proses normalisasi data untuk fitur-fitur yang telah diseleksi di tahap sebelumnya. Normalisasi data diperlukan pada penelitian ini untuk mempersiapkkan *range* masukan yang lebih stabil ke dalam unit LSTM sehingga perhitungan bobot dan bias dapat lebih mudah dilakukan. Nilai pada set data yang akan digunakan, memiliki rentang antara -1 dan 1.30472E+24. Metode normalisasi yang digunakan pada penelitian ini adalah *z-score* menggunakan persamaan 2 . 21, mempertimbangkan karakter *dataset* yang bersifat fluktuatif dan *z-score* mampu menangani selisih drastis nilai maksimal dan nilai minimal data.

Tahap pertama dari z-score adalah mengurangi data dengan nilai rata-rata data tersebut. Kemudian hasil pengurangan tersebut dibagi dengan nilai standar deviasi dari data fitur tersebut untuk mendapatkan seberapa besar pengaruh data tersebut dibandingkan data lain. Nilai data yang dihasilkan dari perhitungan z-score akan berkisar antara -3 dan 3.

Tabel 3.6 Normalisasi z-score

No	Nama Fitur	Sebelum Normalisasi	Setelah Normalisasi
1	SrcPort	34874	-0.126070622
2	Packet Length Mean	61888888888888900	1.768880555
3	Total FWD Packets	5	0.137985532
4	Total BWD Packets	3	1.53145301
5	Flow Packets/s	6635700066357	-0.899570535
6	Flow Byte/s	4620106171201060	-0.408182526
7	Flow Duration	12056	-0.172066677
8	Flow IAT Mean	17222857142857100	-0.076447262
9	Flow IAT Std	19872323898901600	-0.065474964

3.6 Perhitungan LSTM

Perhitungan manual akan menggunakan 9 fitur, yaitu 9 fitur irisan literatur yang sudah dinormalisasi pada tahap sebelumnya. Tahap ini bertujuan untuk mengolah data masukan untuk menghasilkan keluaran yang dapat diklasifikasikan.

Tahap awal perhitungan LSTM adalah inisialisasi bobot dan bias yang dibutuhkan untuk proses awal belajar LSTM networks. Pada contoh perhitungan ini, akan dilakukan sekali proses *epoch* pada 2 unit LSTM yang menerima masukan di setiap unitnya dan mengeluarkan keluaran setiap unitnya. Nilai – nilai yang digunakan diambil contoh nyata dari dataset yang telah dinormalisasi di data. Nilai bobot dan bias akan diinisialisasi secara acak mempertimbangkan data yang sudah dinormalisasi dan akan diperbaharui sesuai dengan proses belajar. Berikut data nilai inisialisasi bobot dan bias.

$$w_{xz} = \begin{pmatrix} 0.3901 \\ -0.0003 \\ 0.0201 \\ 0.0202 \\ -0.0143 \\ -0.1211 \\ -0.1922 \\ -0.2231 \\ 0.185 \end{pmatrix} \quad w_{xi} = \begin{pmatrix} 0.3002 \\ 0.285 \\ 0.1544 \\ -0.2158 \\ 0.1474 \\ 0.0883 \\ -0.212 \\ 0.01 \\ 0.0054 \end{pmatrix} \quad w_{xf} = \begin{pmatrix} 0.2701 \\ -0.2102 \\ 0.2435 \\ 0.276 \\ -0.097 \\ 0.2001 \\ -0.099 \\ 0.1029 \\ 0.1387 \end{pmatrix} \quad w_{xo} = \begin{pmatrix} 0.12 \\ -0.002 \\ 0.243 \\ 0.182 \\ -0.021 \\ 0.031 \\ -0.01 \\ 0.0202 \\ 0.0411 \end{pmatrix}$$

$$w_{hz} = 0.2116$$
$$w_{hi} = -0.032$$
$$w_{hf} = -0.2911$$
$$w_{ho} = 0.1341$$
$$b_z = 0.0958$$
$$b_i = 0.01327$$
$$b_f = -0.2197$$
$$b_{zo} = -0.142$$

Nilai bobot dan bias dengan notasi *f* akan digunakan pada aktivasi *forget gate*, notasi *i* digunakan untuk menunjukkan aktivasi *input gate*, notasi *c* digunakan untuk menunjukkan aktivasi *memory gate* dan notasi *o* digunakan untuk menunjukkan aktivasi pada *output gate*. Setelah inisialisasi selesai, maka proses belajar dapat mulai dilakukan per unit LSTM dengan menggunaka data yang telah dinormalisasi sebagai masukan.

Berikut merupakan nilai yang diperoleh dari fungsi aktivasi sigmoid dari *forget gate* dengan Persamaan 2 . 6 untuk menentukan jumlah informasi yang akan dilupakan dari memori unit sebelumnya untuk diproses pada unit ini. Nilai ini merupakan hasil fungsi aktivasi sigmoid dari masukan saat ini dan keluaran dari unit sebelumya. Hasil keluaran dari *forget gate* akan digunakan dalam perhitungan memori saat ini.

$$f_1 = \sigma(w_{xf}x_1 + w_{hf}h_0 + b_f)$$

$$f_1 = \sigma(0.1837744 + 0 - 0.2197)$$

$$f_1 = \sigma(-0.035926)$$

$$f_1 = 0.4910196$$

Berikut merupakan nilai yang diperoleh dari fungsi aktivasi *input gate* dengan menggunakan Persamaan 2 . 7 untuk menentukan jumlah informasi yang akan digunakan dari masukan untuk diproses pada unit ini. Nilai ini merupakan hasil fungsi aktivasi sigmoid dari masukan saat ini dan keluaran dari unit sebelumnya.

$$i_1 = \sigma(w_{xi}x_1 + w_{hi}h_0 + b_i)$$

$$i_1 = \sigma(0.4597422 + 0 + 0.01327)$$

$$i_1 = \sigma(0.4730122)$$

$$i_1 = 0.6160965$$

Berikut merupakan nilai yang diperoleh dari fungsi aktivasi *memory gate* dengan menggunakan persamaan 2 . 8. Proses ini melakukan fungsi aktivasi *hyperbolic tangent* terhadap masukan unit saat ini dan keluarandari unit sebelumnya. Nilai ini akan menghasilkan informasi yang akan disimpan untuk menjadi memori unit ini.

$$z_1 = tanh(w_{xc}x_1 + w_{hc}h_0 + b_c)$$

$$z_1 = tanh(0.41364546 + 0 + 0.0958)$$

$$z_1 = tanh(0.50944546)$$

$$z_1 = 0.46951301$$

Berikut merupakan perhitungan nilai memori baru dengan menggunakan persamaan 2 . 9. Nilai ini diperoleh dari proses penjumlahan antara memori lama yang telah disaring oleh *forget gate* dengan nilai kandidat memori baru dari *memory gate* yang telah disaring oleh *input gate*. Nilai ini menjadi nilai memori untuk unit saat ini yang akan digunakan dalam proses dan unit selanjutnya.

$$c_1 = f_1 c_0 + i_1 z_1$$

$$c_1 = (0.49101956 * 1 + 0.6160965 * 0.46951301)$$

$$c_1 = 0.780285$$

Berikut merupakan proses perhitungan pada *output gate*. Proses ini melakukan aktivasi sigmoid terhadap masukan saat ini dan keluaran dari unit sebelumnya dengan menggunakan Persamaan 2 . 10. Hasil proses aktivasi *output gate* ini akan menentukan jumlah besaran dari masukan saat ini yang akan dijadikan keluaran untuk unit ini.

$$o_1 = \sigma(w_{xo} x_1 + w_{ho} h_0 + b_o)$$

$$o_1 = \sigma(0.59742 + 0 - 0.142)$$

$$o_1 = \sigma(-0.45542)$$

$$o_1 = 0.611927$$

Berikut merupakan proses perhitungan nilai keluaran yang akan dijadikan sebagai keluaran untuk unit LSTM ini. Nilai ini didapat dari hasil perkalian nilai *output gate* dengan hasil aktivasi *hyperbolic tangent* terhadap nilai memori saat ini menggunakan Persamaan 2 . 11. Nilai keluaran inilah yang bersamaan dengan nilai memori akan diteruskan kepada unit LSTM berikutnya.

$$h_1 = o_1 * tanh(c_1)$$

$$h_1 = 0.611927 * tanh(0.78028486)$$

$$h_1 = 0.611297 * 0.65287$$

$$h_1 = 0.399509$$

Berikut merupakan perhitungan pada unit LSTM ke-2 dimana nilai yang diproses sebagai masukan adalah nilai dari titik data ke-2, nilai memori dan keluaran dari unit LSTM ke-1.

$$f_2 = \sigma(w_{xf} x_2 + w_{hf} h_1 + b_f)$$

$$f_2 = \sigma(0.0925086 - 0.116297 - 0.2197)$$

$$f_2 = \sigma(-0.243488)$$

$$
\begin{aligned}
f_2 &= 0.4394269 \\
\\
i_2 &= \sigma(w_{xi}x_2 + w_{hi}h_1 + b_i) \\
i_2 &= \sigma(0.2314257 - 0.0127843 + 0.01327) \\
i_2 &= \sigma(0.2319115) \\
i_2 &= 0.5577194 \\
\\
z_2 &= tanh(w_{xc}x_2 + w_{hc}h_1 + b_c) \\
z_2 &= tanh(0.2082215 + 0.08453609 + 0.0958) \\
z_2 &= tanh(0.38855759) \\
z_2 &= 0.37011607 \\
\\
c_2 &= f_2c_1 + i_2z_2 \\
c_2 &= (0.4394269 * 0.78028486 + 0.5577194 * 0.37011607) \\
c_2 &= 0.549299 \\
\\
o_2 &= \sigma(w_{xo}x_2 + w_{ho}h_1 + b_o) \\
o_2 &= \sigma(0.30073 + 0.053574 - 0.142) \\
o_2 &= \sigma(-0.212304) \\
o_2 &= 0.552878 \\
\\
h_2 &= o_2 * tanh(c_2) \\
h_2 &= 0.552878 * tanh(0.549299) \\
h_2 &= 0.586142 * 0.499995 \\
h_2 &= 0.276436
\end{aligned}
$$

Dengan selesainya dilakukan sekali laju proses *forward propagation*, selanjutnya akan dilakukan proses pembelajaran dengan *backward propagation* untuk mendapatkan nilai bobot

dan bias yang baru untuk dipakai pada proses iterasi selanjutnya. Langkah perhitungan *backward propagation* dilakukan berkebalikan dari langkah *forward propagation* dengan mencari nilai selisih setiap langkahnya dari titik unit terakhir.

Berikut merupakan proses perhitungan pertama dari proses backward propagation yaitu selisih hasil keluaran unit 2 dengan nilai asli titik data 2 dengan Persamaan 2 . 12. Nilai tersebut merupakan *error* yang dihasilkan pada tahap *forward propagation*, dan nilai ini yang akan menentukan perhitungan tahap-tahap selanjutnya pada proses *backward propagation*.

$$\delta y^2 = \Delta^2 + W_{hc}\delta z^3 + W_{hi}\delta i^3 + W_{hf}\delta f^3 + W_{ho}\delta o^3$$

$$\delta y^2 = (0.276436 + 0 + 0)$$

$$\delta y^2 = 0.276436$$

Berikut merupakan proses perhitungan nilai selisih dari aktivasi *Output Gate* dengan menggunakan persamaan 2 . 13. Nilai ini akan diakumulasikan dengan nilai selisih *gate* lainnya.

$$\delta \bar{o}^2 = \delta y^2 * tanh(c^2) * \sigma(\bar{o}^2)$$

$$\delta \bar{o}^2 = (0.276436 * tanh(0.549299) * 0.447122$$

$$\delta \bar{o}^2 = 0.034168$$

Berikut merupakan proses perhitungan nilai selisih dari memori unit saat ini dengan menggunakan persamaan 2 . 14. Nilai ini akan digunakan untuk perhitungan selisih *gate* lainnya.

$$\delta c^2 = \delta y^2 * o^2 * tanh'(c^2) + \delta c^3 * f^3$$

$$\delta c^2 = (0.276436 * 0.5528776 * 0.7500053 + 0)$$

$$\delta^2 = 0.1146272$$

Berikut merupakan proses perhitungan nilai selisih dari nilai aktivasi *forget gate* dengan menggunakan persamaan 2 . 15. Nilai ini akan diakumulasikan dengan selisih *gate* lainnya untuk mempengaruhi bobot dan bias.

$$\delta\bar{f}^2 = \delta c^2 * c^1 * \sigma(\bar{f}^2)$$

$$\delta\bar{f}^2 = (0.1146272) * 0.780285 * 0.5605731$$

$$\delta\bar{f}^2 = 0.022032$$

Berikut merupakan proses perhitungan nilai selisih dari nilai aktivasi *input gate* dengan menggunakan persamaan 2 . 16. Nilai ini akan diakumulasikan dengan selisih *gate* lainnya untuk mempengaruhi bobot dan bias.

$$\delta\bar{i}^2 = \delta c^2 * z^2 * \sigma(\bar{i}^2)$$

$$\delta\bar{i}^2 = (0.01146272) * 0.549299 * 0.4422806$$

$$\delta\bar{i}^2 = 0.015531$$

Berikut merupakan proses perhitungan nilai selisih dari nilai aktivasi *memory gate* dengan menggunakan persamaan 2 . 17. Nilai ini akan diakumulasikan dengan selisih *gate* lainnya untuk mempengaruhi bobot dan bias.

$$\delta\bar{z}^2 = \delta c^2 * i^2 * tanh'(\bar{z}^2)$$

$$\delta\bar{z}^2 = (0.01146272) * 0.5577194 * 0.69827056$$

$$\delta\bar{z}^2 = 0.04464$$

Dengan terhitungnya semua nilai selisih aktivasi pada unit LSTM ke-2 maka perhitungan akan dilanjutkan dengan langkah yang sama dengan sebelumnya untuk unit LSTM ke-1. Berikut merupakan perhitungan selisih aktivasi unit LSTM ke-1.

$$\delta y^1 = \Delta^1 + W_{hc}\delta z^2 + W_{hi}\delta i^2 + W_{hf}\delta f^2 + W_{ho}\delta o^2$$

$$\delta y^1 = (0.3995089 - 1) + 0.007117$$

$$\delta y^1 = -0.59337$$

$$\delta\bar{o}^1 = \delta y^1 * tanh(c^1) * \sigma(\bar{o}^1)$$

$$\delta\bar{o}^1 = (-0.59337 * tanh(0.780285) * 0.388073$$

$$\delta\bar{o}^1 = -0.092$$

$$\delta c^1 = \delta y^1 * o^1 * tanh'(c^1) + W_{co} * \delta \bar{o}^1 + W_{ci} * \delta \bar{i}^1 + W_{cf} * \delta \bar{f}^2 + \delta c^2 * f^2$$

$$\delta c^1 = -0.59337 * 0.6119271 * 0.0503702 + 0.007117$$

$$\delta c^1 = -0.157963$$

$$\delta \bar{f}^1 = \delta c^1 * c^0 * \sigma(\bar{f}^1)$$

$$\delta \bar{f}^1 = (-0.157963) * 1 * 0.508980443$$

$$\delta \bar{f}^1 = -0.03948$$

$$\delta \bar{i}^1 = \delta c^1 * z^1 * \sigma(\bar{i}^1)$$

$$\delta \bar{i}^1 = (-0.157963) * 0.7802849 * 0.383903$$

$$\delta \bar{i}^1 = -0.02915$$

$$\delta \bar{z}^1 = \delta c^1 * i^1 * tanh'(\bar{z}^1)$$

$$\delta \bar{z}^1 = (-0.157963) * 0.6160965 * 0.3911555$$

$$\delta \bar{z}^1 = -0.038067$$

Setelah selesai dilakukan proses perhitungan selisih untuk seluruh unit LSTM, maka gradien atau selisih nilai bias dan bobot dapat dihitung dengan menggunakan Persamaan 2 . 18 untuk bobot aktivasi masukan titik data, Persamaan 2 . 19 untuk bobot aktivasi keluaran unit sebelumnya dan Persamaan 2 . 20 untuk selisih nilai bias.

$$\delta W_{xf} = \sum_{t=1}^{2} (\delta \bar{f}^t x^t)$$

$$\delta W_{xf} = (\delta \bar{f}^1 x^1) + (\delta \bar{f}^2 x^2)$$

$$\delta W_{xf} = \begin{pmatrix} 0.021961832 \\ -0.030859411 \\ -0.002407258 \\ -0.026717314 \\ 0.035698826 \\ 0.013129338 \\ 0.002994653 \\ 0.001333662 \\ 0.00114225 \end{pmatrix}$$

$$\delta W_{xi} = \sum_{t=1}^{2} (\delta \bar{i}^t x^t)$$

$$\delta W_{xi} = (\delta \bar{i}^1 x^1) + (\delta \bar{i}^2 x^2)$$

$$\delta W_{xi} = \begin{pmatrix} \\ 0.015648532 \\ -0.024094571 \\ -0.001879552 \\ -0.020860483 \\ 0.026355762 \\ 0.009795475 \\ 0.002338725 \\ 0.001041305 \\ 0.000891852 \end{pmatrix}$$

$$\delta W_{xo} = \sum_{t=1}^{2} (\delta \bar{o}^t x^t)$$

$$\delta W_{xo} = (\delta \bar{o}^1 x^1) + (\delta \bar{o}^2 x^2)$$

$$\delta W_{xo} = \begin{pmatrix} 0.037937999 \\ -0.102291188 \\ -0.007979456 \\ -0.088561179 \\ 0.083044526 \\ 0.032922119 \\ 0.009939179 \\ 0.004420784 \\ 0.003786286 \end{pmatrix}$$

$$\delta W_{xz} = \sum_{t=1}^{2} (\delta \bar{z}^t x^t)$$

$$\delta W_{xz} = (\delta \bar{z}^1 x^1) + (\delta \bar{z}^2 x^2)$$

$$\delta W_{xz} = \begin{pmatrix} 0.039212634 \\ 0.011626612 \\ 0.00090696 \\ 0.010066033 \\ 0.034620314 \\ 0.009490663 \\ -0.00114551 \\ -0.000502509 \\ -0.000430376 \end{pmatrix}$$

$$\delta W_{hf} = \sum_{t=1}^{1} (\delta \bar{f}^t y^t)$$

$$\delta W_{hf} = (\delta \bar{f}^1 x^1)$$

$$\delta W_{hf} = 0.0088021$$

$$\delta W_{hi} = \sum_{t=1}^{1}(\delta \bar{i}^t y^t)$$
$$\delta W_{hi} = (\delta \bar{i}^1 x^1)$$
$$\delta W_{hi} = 0.006204927$$

$$\delta W_{ho} = \sum_{t=1}^{1}(\delta \bar{o}^t y^t)$$
$$\delta W_{ho} = (\delta \bar{o}^1 x^1)$$
$$\delta W_{ho} = 0.013650283$$

$$\delta W_{hz} = \sum_{t=1}^{1}(\delta \bar{z}^t y^t)$$
$$\delta W_{hz} = (\delta \bar{z}^1 x^1)$$
$$\delta W_{hz} = 0.017834203$$

$$\delta b_f = \sum_{t=1}^{2}(\delta \bar{f}^t)$$
$$\delta b_f = (\delta \bar{f}^1) + (\delta \bar{f}^2)$$
$$\delta b_f = -0.017445729$$

$$\delta b_i = \sum_{t=1}^{2}(\delta \bar{i}^t)$$
$$\delta b_i = (\delta \bar{i}^1) + (\delta \bar{i}^2)$$
$$\delta b_i = -0.013621367$$

$$\delta b_o = \sum_{t=1}^{2}(\delta \bar{o}^t)$$
$$\delta b_o = (\delta \bar{o}^1) + (\delta \bar{o}^2)$$
$$\delta b_o = -0.057828205$$

$$\delta b_z = \sum_{t=1}^{2}(\delta \bar{z}^t)$$
$$\delta b_z = (\delta \bar{o}^1) + (\delta \bar{z}^2)$$

$$\delta b_z = 0.006572864$$

Setelah didapatkan nilai selisih untuk setiap bobot dan bias pada masing-masing *gate*, berikut merupakan hasil perhitungan nilai bobot dan bias yang baru dengan mengurangkan bobot dan bias asal dengan selisihnya yang telah dikalikan dengan nilai *learning rate*.

$$W_{xf} = W_{xf} - \delta W_{xf}$$

$$W_{xf} = \begin{pmatrix} 0.117803817 \\ 0.001085941 \\ 0.243240726 \\ 0.184671731 \\ -0.024569883 \\ 0.029687066 \\ -0.010299465 \\ 0.020066634 \\ 0.040985775 \end{pmatrix}$$

$$W_{xi} = W_{xi} - \delta W_{xi}$$

$$W_{xi} = \begin{pmatrix} 0.298635147 \\ 0.287409457 \\ 0.154587955 \\ -0.213713952 \\ 0.144764424 \\ 0.087320453 \\ -0.212233872 \\ 0.009895869 \\ 0.005310815 \end{pmatrix}$$

$$W_{xo} = W_{xo} - \delta W_{xo}$$

$$W_{xo} = \begin{pmatrix} 0.3863062 \\ 0.009929119 \\ 0.020897946 \\ 0.029056118 \\ -0.022604453 \\ -0.124392212 \\ -0.193193918 \\ -0.223542078 \\ 0.184621371 \end{pmatrix}$$

$$W_{xz} = W_{xz} - \delta W_{xz}$$

$$W_{xz} = \begin{pmatrix} 0.266178737 \\ -0.211362661 \\ 0.243409304 \\ 0.274993397 \\ -0.100462031 \\ 0.199150934 \\ -0.098885449 \\ 0.102950251 \\ 0.138743038 \end{pmatrix}$$

$$W_{hf} = W_{hf} - \delta W_{hf}$$

$$W_{hf} = -0.29198021$$

$$W_{hi} = W_{hi} - \delta W_{hi}$$

$$W_{hi} = -0.032620493$$

$$W_{ho} = W_{ho} - \delta W_{ho}$$

$$W_{ho} = 0.132734972$$

$$W_{hz} = W_{hz} - \delta W_{hz}$$

$$W_{hz} = 0.20981658$$

$$W_{bf} = W_{bf} - \delta W_{bf}$$

$$W_{bf} = -0.217955427$$

$$W_{bi} = W_{bi} - \delta W_{bi}$$

$$W_{bi} = 0.014632137$$

$$W_{bo} = W_{bo} - \delta W_{bo}$$

$$W_{bo} = -0.136217179$$

$$W_{bz} = W_{bz} - \delta W_{bz}$$

$$W_{bz} = 0.095142714$$

Dengan diperolehnya seluruh nilai bobot dan bias yang baru untuk proses perhitungan iterasi selanjutnya, maka selesailah 1 iterasi perhitungan LSTM dengan konfigurasi 2 *hidden layer*.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Lingkungan Implementasi

Pada lingkungan implementasi, akan dijelaskan mengenai perangkat keras dan perangkat lunak yang digunakan untuk membangun sistem penerapan *Recurrent Neural Network* dengan *Long Short Term Memory* untuk mendeteksi *malware botnet* berdasarkan fitur *network traffic*.

4.1.1 Spesifikasi Perangkat Keras

Spesifikasi komputer yang digunakan dalam pembangunan sistem penerapan *Recurrent Neural Network* dengan *Long Short Term Memory* untuk mendeteksi *malware botnet* berdasarkan fitur *network traffic* adalah sebagai berikut:

1. Laptop dengan *processor* Intel(R) Core(TM) i7-4710HQ CPU 2.50GHz (8CPUs), 2.5GHz
2. *Harddisk* dengan kapasitas 1 TB
3. RAM 8 GB

4.1.2 Lingkungan Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pembangunan sistem penerapan *Recurrent Neural Network* dengan *Long Short Term Memory* untuk mendeteksi *malware botnet* berdasarkan fitur *network traffic* adalah sebagai berikut:

1. Sistem Operasi: Windows 10 Pro 64-bit
2. *Development Tool*: NetBeans IDE 8.2 64-bit
3. *Development Kit*: Java Development Kit (JDK) versi 1.8.0_171

4.2 Implementasi Perangkat Lunak

Bagian ini akan menjelaskan mengenai implementasi sistem untuk pengidentifikasian *botnet* menggunakan LSTM berdasarkan fitur *network traffic*.

4.2.1 Daftar *Class* dan Metode

Berikut ini akan dijelaskan mengenai *class* dan metode serta penjelasan cara kerja program.

4.2.1.1 *Class* LSTM_Model

Class LSTM_Model merupakan objek yang akan mengolah data, melakukan perhitungan LSTM, dan membentuk model yang akan disimpan sebagai hasil proses pembelajaran dan akan digunakan untuk proses pengujian sesuai konfigurasi *class*. Berikut merupakan daftar atribut pada *class* LSTM.

Tabel 4.1 Daftar atribut pada *Class* LSTM_Model

No	Atribut	Tipe Data	Keterangan
1	inputWidth	int	Menampung ukuran matriks pada 1 titik data
2	learningRate	double	Menampung nilai *learning rate* yang akan digunakan
3	times	int	Menampung jumlah *forward propagation* yang sudah dilakukan
4	cVecs, hVecs, fVecs, iVecs, oVecs, ctVecs	double[]	Menampung hasil perhitungan setiap *gate*
5	triOut, deltaOut, deltaOutmin1	double[]	Menampung nilai selisih keluaran
6	deltaState, deltaC, deltaF, deltaI, deltaO, deltaH	double[]	Menampung nilai selisih setiap *gate*
7	deltaX, deltas	double[][]	Menampung nilai selisih dalam kesatuan matriks untuk perhitungan gradien.
8	wfh, bf, wih, bi, woh, bo, wch, wc	double	Menampung nilai bobot aktivasi keluaran unit sebelumnya dan bias
9	whx, wix, wox, wcx	double[]	Menampung nilai bobot aktivasi titik data saat ini
10	weightInput	double[][]	Menampung matriks bobot aktivasi masukan
11	weightInputMin1, bias	double[]	Menampung matriks bobot aktivasi keluaran unit sebelumnya

12	xGrad	double[][]	Menampung gradien bobot aktivasi masukan
13	hGrad, bGrad	double[]	Menampung gradien bobot aktivasi keluaran unit sebelumnya dan bias
14	forwardTimes	int	Menampung jumlah maksimal *forward propagation*

Berikut merupakan daftar metode pada *Class* LSTM_Model.

Tabel 4.2 Daftar Metode pada *Class* LSTM_Model

No	**Metode**	**Masukan**		**Keluaran**	**Keterangan**
		Tipe	**Variabel**		
1	Initialize	-	-	-	Digunakan untuk menginisialisasi seluruh array dan variabel
2	Forward	double[]	x	-	Digunakan untuk melakukan tahap forward propagation dalam proses LSTM
3	CalculateGate	double[], double[], double, double, Activator	x, wx, wh, b, activator	double	Digunakan untuk perhitungan aktivasi setiap gate dalam proses LSTM
4	Backward	double[], double, int	x, deltaH, backStep	-	Digunakan untuk melakukan tahap backward propagation dalam proses LSTM
5	CalculateGrad	double[][], int	input, indexInput	-	Digunakan untuk perhitungan gradien untuk backward propagation LSTM
6	UpdateWeight	-	-	-	Digunakan untuk menghitung weight baru dan memperbaharui model LSTM
7	ResetStates	-	-	-	Digunakan untuk menginisialisasi ulang variabel dalam LSTM

8	ReadWeight FromFile	-	-	-	Digunakan untuk membaca weight dari file dan disimpan ke dalam variabel lokal
9	GetRecord FromLine	String	line	List<double>	Digunakan untuk membaca setiap baris dari file weight dan merubah string menjadi double
10	WriteWeight ToFile	-	-	-	Digunakan untuk menyimpan nilai weight ke dalam bentuk file txt.

4.2.1.2 *Class* Matrix

Class Matrix merupakan objek yang melakukan perhitungan yang dibutuhkan berkaitan dengan pengolahan matriks. Berikut merupakan daftar metode pada *Class* Matrix.

Tabel 4.3 Daftar Metode pada *Class* Matrix

No	Metode	Masukan		Keluaran	Keterangan
		Tipe	Variabel		
1	MulMtrx	double[], double[]	x, y	double	Digunakan untuk menghitung dot product dua buah matriks
2	KaliMtrx	double[][], double[]	a, b	double[]	Digunakan untuk mengalikan matriks 2 dimensi dengan matriks 1 dimensi
3	KaliMtrx	double[], double[]	a, b	double[][]	Digunakan untuk mengalikan matriks 1 dimensi dengan matriks 1 dimensi
4	KaliMtrx	double[], double	a, b	double[]	Digunakan untuk mengalikan matriks 1 dimensi dengan koefisien
5	TambahMtrx	double[][], double[][]	a, b	double[][]	Digunakan untuk menjumlahkan 2 buah matriks 2 dimensi
6	TambahMtrx	double[], double[]	a, b	double[]	Digunakan untuk menjumlahkan 2 buah matriks 1 dimensi
7	KurangMtrx	double[], double[]	a, b	double[]	Digunakan untuk mengurangkan 2 buah matriks 1 dimensi

4.2.1.3 *Class* SigmoidActivator

Class SigmoidActivator merupakan objek yang melakukan perhitungan dengan fungsi aktivasi sigmoid. Berikut merupakan daftar metode pada *class* SigmoidActivator.

Tabel 4.4 Daftar Metode pada *Class* SigmoidActivator

No	Metode	Masukan		Keluaran	Keterangan
		Tipe	Variabel		
1	GetInstance	-	-	SigmoidActivator	Digunakan untuk mendapatkan instansi *class*

2	Forward	double	f	double	Digunakan untuk perhitungan aktivasi sigmoid

4.2.1.4 *Class* TanhActivator

Class TanhActivator merupakan objek yang melakukan perhitungan dengan fungsi aktivasi *hyperbolic tangent*. Berikut merupakan daftar metode pada *Class* TanhActivator.

Tabel 4.5 Daftar Metode pada *Class* TanhActivator

No	Metode	Masukan		Keluaran	Keterangan
		Tipe	Variabel		
1	GetInstance	-	-	TanhActivator	Digunakan untuk mendapatkan instansi *class*
2	Forward	double	f	double	Digunakan untuk perhitungan hyperbolic tangent

4.2.1.5 *Class* PCA_Class

Class PCA_Class merupakan objek yang melakukan perhitungan *Principal Component Analysis* untuk tahap *preprocessing* yang bertujuan untuk mereduksi dimensi fitur. Berikut merupakan daftar metode pada *Class* PCA_Class

Tabel 4.6 Daftar Metode pada *Class* PCA_Class

No	Metode	Masukan		Keluaran	Keterangan
		Tipe	Variabel		
1	GetEigenVec	-	-	double[][]	Digunakan untuk mendapatkan array nilai *eigen vector*
2	GetEigenVal	-	-	double	Digunakan untuk mendapatkan nilai *eigen values*
3	PrintDataset	double[]	label	-	Digunakan untuk menampilkan hasil proses PCA pada sistem
4	CountMtrxCovariance	double[][]	x	-	Digunakan untuk menghitung dan menghasilkan matriks kovarian

5	CountCovariance	double[], double[]	x, y	double	Digunakan untuk menghitung nilai kovarian
6	CountEigenVec	double[][]	x	-	Digunakan untuk menghitung dan menghasilkan *eigen vector*
7	CountMean	double[]	x	double	Digunakan untuk menghitung rata-rata dari sebuah array
8	SortEigen	double[][], double[]	eigenVec, eigenVal	-	Digunakan untuk mengurutkan data pada *eigen vector* dari nilai yang terbesar
9	ReduceDimension	double	threshold	-	Digunakan untuk mengurangi jumlah fitur PCA sesuai threshold yang diinginkan
10	CreateDataset	-	-	-	Digunakan untuk mengalikan hasil PCA ke dataset semula

Berikut merupakan daftar atribut pada *class* PCA_Class

Tabel 4.7 Daftar atribut pada *class* PCA_Class

No	Atribut	Tipe Data	Keterangan
1	data	double[][]	Menampung set data masukan
2	eigenVec, eigenVecReduced	double[][]	Menampung nilai dari vektor *eigen*
3	eigenVal	double[]	Menampung nilai dari *eigen value*
4	covMtrx	double[][]	Menampung nilai dari matriks kovarians
5	eSort, eReduced	EigenSort[]	Menampung nilai vektor *eigen* yang sudah diurutkan
6	dataset	double[][]	menampung hasil set data yang baru

4.2.2 Implementasi Pengambilan Set Data

Data *network traffic* didapat dari *dataset* University of Victoria dalam bentuk file pcap. Data tersebut sudah diberi label yang membedakan antara *botnet* dan *benign*.

4.2.3 *Preprocessing* Data

Sebelum data diproses oleh sistem sebagai masukan untuk model LSTM, data akan melalui tahap *preprocessing* terlebih dahulu untuk mendapatkan nilai data yang siap untuk digunakan. Proses ini dilakukan di luar sistem dengan mengolah dataset dalam bentuk *file* pcap dan mengubahnya menjadi *file* csv yang siap digunakan.

4.2.3.1 Pemilihan Fitur

Pemilihan fitur akan dilaksanakan sesuai rancangan yang telah dibahas di Bab 3.5.1 yaitu mengimplementasikan 5 variasi fitur yang akan digunakan sebagai masukan untuk perhitungan LSTM.

Tabel 4.8 Empat variasi fitur yang digunakan

No	Nama Fitur	Jumlah Fitur
1	Fitur Irisan Literatur	9
2	Fitur Peneliti	8
3	Fitur Kombinasi	17
4	Fitur Gabungan Literatur	27
5	Fitur PCA	30

4.2.3.2 Normalisasi Data

Data yang telah melalui *feature selection* kemudian dinormalisasi dengan menggunakan fungsi normalisasi *Z-Score* untuk memperoleh persebaran data yang lebih merata yang akan disimpan dalam berkas eksternal dengan format *.csv sesuai nama fitur. Berikut merupakan contoh data sebelum dinormalisasi.

1921685088	34197	8844	53	17	42903.43422	4289	1
1921685088	39125	8888	53	17	42902.41544	24645	1
1921685088	34950	8844	53	17	42904.22365	11929	1
1921685088	34994	8888	53	17	42902.48669	5086	1
1921685088	45718	9118989199	123	17	42902.3766	152613	1
1921685088	39972	8844	53	17	42903.33584	3905	1
1921685088	38225	8888	53	17	42905.16993	35705	1
1921685088	41742	8844	53	17	42905.36321	4055	1
1921685063	57339	1921685088	53	17	42903.19547	325	1
1921685050	54679	1921685088	53	17	42906.46941	23393	2
1921685019	63022	1921685088	53	17	42904.36066	71159	1
1921685063	61394	1921685088	53	17	42903.40398	189	1
1921685088	55696	8844	53	17	42904.45479	4628	1
1921685088	41091	8844	53	17	42906.27359	4887	1
1921685069	60486	2.39255E+11	1900	17	42902.17416	6035177	6
8601	0	8064	0	0	42903.31115	118766409	20
1921685019	17500	2.55255E+11	17500	17	42905.25141	119999487	5
1921685015	60484	1921685088	53	17	42846.42134	223	1
1921685088	50259	8888	53	17	42907.1448	5858	1
1921685088	58368	8844	53	17	42847.3005	10010835	4

Gambar 4.1 *Data sebelum dinormalisasi*

Berikut merupakan contoh data setelah dinormalisasi.

-0.46682	-0.76279	-0.08598	0.00277	-0.6377	-0.00338	-0.46897	-0.13499	-0.14386	-0.07698
-0.21345	2.90722	-0.08598	0.00277	-0.38674	-0.00333	-0.46848	-0.13499	-0.14386	-0.06044
-0.4281	1.0505	-0.08598	0.00277	0.24214	-0.00338	-0.46879	-0.13499	-0.14386	-0.16797
-0.42584	1.08959	-0.08598	0.00277	1.88203	-0.00339	-0.46895	-0.13499	-0.14386	-0.11834
0.12551	-0.76279	-0.08598	0.00277	-0.024	-0.00338	-0.46536	-0.13499	-0.14386	-0.08525
-0.16991	-0.76279	-0.08598	0.00277	-0.60437	-0.00338	-0.46898	-0.13499	-0.14386	-0.00253
-0.25973	-0.76279	-0.08598	0.00277	-0.56949	-0.00338	-0.46821	-0.13499	-0.14386	-0.12661
-0.07891	-0.24595	-0.08598	0.00277	2.60887	-0.00338	-0.46898	-0.13499	-0.14386	-0.06044
0.72299	0.15797	-0.08598	0.00277	-0.52933	-0.00337	-0.46907	-0.13499	-0.14386	-0.15143
0.58623	-0.76279	-0.05154	0.02488	0.26636	-0.00339	-0.46851	-0.13499	-0.14386	-0.23415
1.01517	0.05591	-0.08598	0.00277	-0.77237	-0.00338	-0.46734	-0.13499	-0.14386	-0.1597
0.93147	0.26872	-0.08598	0.00277	-0.20741	-0.00338	-0.46907	-0.13499	-0.14386	-0.11007
0.63851	0.15363	-0.08598	0.00277	-0.94528	-0.00339	-0.46896	-0.13499	-0.14386	0.32835
-0.11238	0.49891	-0.08598	0.00277	1.99844	-0.00339	-0.46896	-0.13499	-0.14386	-0.25069
0.88478	0.04598	0.08623	-0.01933	-0.25396	-0.00337	-0.32192	-0.13499	-0.14386	0.5517
-2.225	-0.76279	0.56841	-0.01933	0.24751	-0.00339	2.42676	-0.13499	-0.14386	-0.48231
-1.32527	-0.76279	0.05179	-0.01933	-0.67379	-0.00338	2.45683	-0.13498	-0.14386	0.62615
0.88468	-0.76279	-0.08598	0.00277	-0.3247	-0.00338	-0.46907	-0.13499	-0.14386	-0.20106
0.35898	0.49891	-0.08598	0.00277	-0.7285	-0.00338	-0.46893	-0.13499	-0.14386	-0.25069
0.77589	-0.76279	0.01735	-0.01933	-0.68622	-0.00337	-0.22499	-0.13499	-0.14386	-0.21761

Gambar 4.2 *Data sesudah dinomalisasi*

4.2.4 Implementasi Metode LSTM

Pada tahap ini akan dibahas mengenai langkah-langkah implementasi metode LSTM untuk mendeteksi *malware botnet* berdasarkan fitur-fitur dari *network traffic*.

1. Membaca dokumen csv yang akan diolah
   Dokumen yang sudah ditentukan dari masukan akan divalidasi ketersediaan dokumen pada URL yang bersangkutan. Apabila benar, dokumen akan dibaca dan dipisahkan sesuai dengan delimiter yang sudah didefinisikan. Hasil ini akan dimasukkan ke dalam variabel sistem dengan pola 70% untuk pembelajaran dan 30% dari keseluruhan data untuk pengujian.

2. Inisialisasi model LSTM
   Konfigurasi model LSTM yang sudah ditentukan saat proses input akan digunakan untuk menginisialisasi model LSTM. Proses ini mencakup mendefinisikan jumlah layer, learning rate, menentukan nilai bobot dan bias awal secara *random*, serta nilai memori awal yang akan digunakan selama proses belajar dan pengujian berlangsung.

3. Proses pembelajaran
   Proses pembelajaran menggunakan 70% data keseluruhan sebagai masukan kepada model LSTM yang sudah diinisialisasi sebanyak jumlah iterasi yang sudah ditentukan pada tahap masukan. Proses belajar ini bertujuan untuk mengatur nilai bobot dan bias pada model LSTM dengan data yang diproses untuk memberikan akurasi hasil keluaran yang lebih baik. Dalam 1 iterasi, akan dilakukan *forward propagation* sebanyak jumlah unit model LSTM dan *backward propagation* untuk setiap unit hasil forward propagation. Hasil iterasi ini akan memperbaharui nilai bobot dan bias untuk digunakan pada iterasi selanjutnya sampai jumlah *epoch* tercapai.

4. Proses pengujian
   Proses pengujian menggunakan 30% data keseluruhan sebagai masukan kepada model LSTM dengan bobot dan bias yang sudah diatur pada tahap pembelajaran. Dalam 1 iterasi, akan dilakukan forward propagation sebanyak jumlah layer dan hasil dari layer terakhir kemudian akan disimpulkan sebagai keluaran dari 1 iterasi.

5. Validasi model LSTM
   Akurasi sistem akan diukur dan disimpulkan dalam bentuk matriks yang berisi informasi hasil pengujian berupa akurasi, presisi, *recall*, dan F-Score. Hasil akurasi tersebut akan dibandingkan dan dianalisis untuk memperoleh konfigurasi model LSTM yang paling optimal terhadap data masukan.

4.2.5 Berkas Eksternal

Sistem ini menggunakan berkas eksternal sebagai masukan dalam bentuk *.csv agar sistem dapat menjalankan fungsinya. Data di dalam berkas *.csv memiliki format berupa nilai data yang dimulai dari baris pertama dan tiap nilai data dipisahkan oleh tanda baca titik koma.

Untuk memudahkan proses pembelajaran dan pengujian, hasil perubahan nilai bobot dan bias dari satu iterasi LSTM akan disimpan dalam bentuk *.txt. Hasil ini akan digunakan kembali untuk iterasi LSTM berikutnya.

4.3 Skenario Pengujian

Bagian ini akan menjelaskan pengujian yang dilakukan pada sistem perangkat lunak yang telah dibuat. Hasil akhir dari pengujian ini adalah melihat pengaruh parameter masukan yang diberikan terhadap hasil klasifikasi.

4.3.1 Data Pengujian

Dalam penelitian ini digunakan *dataset* sesuai yang sudah dijelaskan pada Bab 3.4 dengan jumlah 87854 data yang mencakup 71835 data *benign* dan 16019 data *botnet*. *Dataset* kemudian dipisahkan menjadi 70% data latih dengan jumlah 61498 data yang mencakup 50369 data *benign* dan 11129 data *botnet* serta 30% data uji dengan jumlah 26356 data yang mencakup 21466 data *benign* dan 4890 data *botnet*.

Tabel 4.9 Jumlah data yang akan digunakan

Data	**Benign**	**Botnet**	**Total**
Sumber	71835	16019	87854
Data Latih	50369	11129	61498
Data Uji	21466	4890	26356

4.3.2 Analisis Parameter Pengujian

Berikut adalah pengujian yang akan dilakukan untuk mendapatkan hasil-hasil pengujian yang diinginkan.

1. **Variasi fitur**. Fitur yang akan digunakan dalam pengujian berjumlah 5 sesuai yang dijelaskan pada Bab 3.5.1 yaitu fitur irisan literatur, fitur peneliti, fitur kombinasi, fitur gabungan literatur dan fitur *PCA*.

2. **Variasi jumlah unit LSTM**. Unit LSTM yang akan digunakan dalam pengujian adalah 2 (minimal unit LSTM), 5, 10, 20, 100, 200, 500, 1000, 2000, dan 4000. Variasi ini bertujuan untuk melihat perubahan pada hasil klasifikasi.

3. **Variasi jumlah *epoch* yang dilakukan**. Jumlah epoch yang digunakan dalam pengujian adalah 1, 2, 3, 10 dan 500. Parameter ini bertujuan untuk melihat pengaruh banyaknya iterasi pembelajaran dataset terhadap akurasi klasifikasi.

4. **Variasi nilai *learning rate* pada unit LSTM**. Nilai *learning rate* yang digunakan dalam pengujian adalah 0.01, 0.1 dan 0.5. Parameter ini bertujuan untuk melihat pengaruh *learning rate* pada bobot terhadap kualitas model klasifikasi.

Tabel 4.10 Parameter Model LSTM

Nama Parameter	**Jumlah Variasi**	**Keterangan**
Fitur	5	Fitur irisan literatur, fitur peneliti, fitur kombinasi, fitur gabungan literatur dan fitur PCA
Unit LSTM	10	2, 5, 10, 20, 100, 200, 500, 1000, 2000, 4000
Epoch	5	1, 2, 3, 10, 500
Learning Rate	3	0.01, 0.1, 0.5

Daftar skenario pengujian tercantum pada Lampiran C Tabel C-1.

4.3.3 Skenario Analisis Hasil Pengujian

Data-data hasil pengujian akan disajikan dalam bentuk tabel yang berisi informasi mengenai parameter pengujian, nilai *true positive*, *true negative*, *false positive*, *false negative*, serta nilai akurasi dan *recall* dari masing-masing pengujian. Akurasi dan *recall* menjadi standar utama penilaian keberhasilan sistem dalam mengklasifikasikan *malware botnet* dan aplikasi *benign*.

Akurasi sebagaimana yang sudah dijelaskan pada Bab 2, menunjukkan sebarapa baik sistem dalam memisahkan aktivitas *malware botnet* maupun *benign* dengan menjumlahkan nilai *true positive* dan *true negative* dibagi total data uji. *Recall* menunjukkan keberhasilan sistem dalam mendeteksi seluruh *malware botnet* yaitu dengan membagi nilai *true positive* terhadap total dari *true positive* dan *false negative*. Nilai akurasi yang tinggi menandakan sistem berhasil memisahkan dengan baik setiap aktivitas berdasarkan label sesungguhnya (*true positive* dan *true negative*). Nilai *recall* yang tinggi menandakan keberhasilan sistem dalam mendeteksi aktivitas *malware botnet* (*true positive*) dan meminimalisir kesalahan sistem dalam mengklasifikasikan *malware botnet* sebagai aplikasi *benign* (false negative).

Oleh karena itu, dapat disimpulkan bahwa keberhasilan penelitian ini dianalisis dalam kemampuan sistem menghasilkan nilai akurasi dan *recall* yang baik.

4.4 Pengujian

Bab ini akan menganalisa hasil pengujian sesuai skenario pengujian yang sudah dibahas sebelumnya. Hasil pengujian akan dianalisa dalam bentuk tabel berdasarkan variasi jumlah unit LSTM disertai dengan hasil analisis berupa tabel informasi hasil akurasi dan

Recall, serta tabel rangkuman hasil pengujian tertinggi terhadap setiap metode pemilihan fitur. Berikut adalah hasil pengujian dengan variasi *epoch*, *learning rate* dan variasi jumlah unit LSTM terhadap lima variasi fitur.

4.4.1 Pengujian Fitur Irisan Literatur

Berikut merupakan hasil pengujian terhadap fitur irisan literatur.

4.4.1.1 Pengujian 2 Unit LSTM

Unit LSTM terkecil untuk dapat dilakukan sebuah iterasi adalah 2. Berikut hasil pengujian untuk 2 unit LSTM.

Tabel 4.11 Hasil pengujian terhadap 2 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3147	18669	2797	1743	82.77432%	64.35583%
2	1	0.5	3623	18404	3062	1267	83.57489%	74.08998%
3	2	0.1	3266	18691	2775	1624	83.30930%	66.78937%
4	2	0.5	3502	18744	2722	1388	84.40582%	71.61554%
5	3	0.1	3254	18834	2632	1636	83.80634%	66.54397%
6	3	0.5	3427	19255	2211	1463	86.06010%	70.08180%
7	10	0.1	3137	19903	1563	1753	87.41842%	**74.08998%**
8	10	0.5	3170	19934	1532	1720	**87.66125%**	64.82618%

4.4.1.2 Pengujian 5 Unit LSTM

Pengujian berikut menggunakan 5 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 5 unit LSTM.

Tabel 4.12 Hasil pengujian terhadap 5 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3892	17144	4319	998	79.82392%	79.59100%
2	1	0.5	3900	16430	5033	990	77.14491%	79.75460%
3	2	0.1	3573	17897	3566	1317	81.47080%	73.06748%
4	2	0.5	3911	17313	4150	979	80.53732%	79.97955%
5	3	0.1	3313	18540	2923	1577	82.92414%	67.75051%
6	3	0.5	3924	17201	4262	966	80.16165%	80.24540%
7	10	0.1	3398	18897	2566	1492	**84.60137%**	69.48875%
8	10	0.5	3989	17111	4352	901	80.06678%	**81.57464%**

4.4.1.3 Pengujian 10 Unit LSTM

Pengujian berikut menggunakan 10 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 10 unit LSTM.

Tabel 4.13 Hasil pengujian terhadap 10 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3928	17040	4418	962	79.58099%	**80.32720%**
2	1	0.5	3694	17934	3524	1196	82.08592%	75.54192%
3	2	0.1	3811	17836	3622	1079	82.15803%	77.93456%
4	2	0.5	3787	18819	2639	1103	85.79778%	77.44376%
5	3	0.1	3618	18411	3047	1272	83.60786%	73.98773%
6	3	0.5	3799	18899	2559	1091	86.14695%	77.68916%
7	10	0.1	3746	18454	3004	1144	84.25687%	76.60532%
8	10	0.5	3562	19179	2279	1328	**86.31015%**	72.84254%

4.4.1.4 Pengujian 20 Unit LSTM

Pengujian berikut menggunakan 20 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 20 unit LSTM.

Tabel 4.14 Hasil pengujian terhadap 20 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3824	17713	3738	1063	81.77158%	**78.24841%**
2	1	0.5	3699	18386	3065	1188	83.85223%	75.69061%
3	2	0.1	3723	18261	3190	1164	83.46875%	76.18171%
4	2	0.5	3652	19011	2440	1235	86.04678%	74.72887%
5	3	0.1	3733	18197	3254	1154	83.26372%	76.38633%
6	3	0.5	3777	18907	2544	1110	**86.12651%**	77.28668%
7	10	0.1	3797	18301	3150	1090	83.90158%	77.69593%
8	10	0.5	3627	19038	2413	1260	86.05437%	74.21731%

4.4.1.5 Pengujian 100 Unit LSTM

Pengujian berikut menggunakan 100 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 100 unit LSTM.

Tabel 4.15 Hasil pengujian terhadap 100 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3302	18455	2933	1568	82.85855%	67.80287%
2	1	0.5	3317	18732	2656	1553	83.97059%	68.11088%
3	2	0.1	3254	19471	1917	1616	86.54505%	66.81725%
4	2	0.5	3000	19082	2306	1870	84.09627%	61.60164%
5	3	0.1	3124	19669	1719	1746	86.80402%	64.14784%
6	3	0.5	3332	18248	3140	1538	82.18447%	**68.41889%**
7	10	0.1	3083	19776	1612	1787	**87.05537%**	63.30595%
8	10	0.5	1124	20926	462	3746	83.97440%	23.08008%

4.4.1.6 Pengujian 200 Unit LSTM

Pengujian berikut menggunakan 200 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 200 unit LSTM.

Tabel 4.16 Hasil pengujian terhadap 200 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	2929	19752	1552	1925	86.70769%	**60.34199%**
2	1	0.5	2445	20122	1182	2409	86.27188%	50.37083%
3	2	0.1	2694	19968	1336	2160	86.63506%	55.50062%
4	2	0.5	2491	20098	1206	2363	86.35599%	51.31850%
5	3	0.1	2700	19928	1376	2154	86.50508%	55.62423%
6	3	0.5	2491	20114	1190	2363	86.41715%	51.31850%
7	10	0.1	2720	20062	1242	2134	**87.09381%**	56.03626%
8	10	0.5	2543	20063	1241	2311	86.42098%	52.38978%

4.4.1.7 Pengujian 500 Unit LSTM

Pengujian berikut menggunakan 500 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 500 unit LSTM.

Tabel 4.17 Hasil pengujian terhadap 500 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	2992	18894	2166	1806	84.63918%	62.35932%
2	1	0.5	2701	19245	1815	2097	84.87121%	56.29429%
3	2	0.1	2978	18892	2168	1820	84.57730%	62.06753%
4	2	0.5	3299	18514	2546	1499	84.35687%	**68.75782%**
5	3	0.1	2790	19162	1898	2008	84.89442%	58.14923%
6	3	0.5	1745	20132	928	3053	84.60437%	36.36932%
7	10	0.1	2914	19022	2038	1884	84.83254%	60.73364%
8	10	0.5	2569	19643	1417	2229	**85.89991%**	53.54314%

4.4.1.8 Pengujian 1000 Unit LSTM

Pengujian berikut menggunakan 1000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 1000 unit LSTM.

Tabel 4.18 Hasil pengujian terhadap 1000 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	2114	19459	1182	2603	85.07374%	44.81662%
2	1	0.5	2089	19712	929	2628	85.97286%	44.28662%
3	2	0.1	2006	19505	1136	2711	84.82924%	42.52703%
4	2	0.5	2667	19050	1591	2050	85.61461%	56.54017%
5	3	0.1	3089	18361	2280	1628	84.58868%	**65.48654%**
6	3	0.5	2189	19647	994	2528	**86.11089%**	46.40661%
7	10	0.1	2625	18781	1860	2092	84.41517%	55.64978%
8	10	0.5	2742	19026	1615	1975	85.84273%	58.13017%

4.4.1.9 Pengujian 2000 Unit LSTM

Pengujian berikut menggunakan 2000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 2000 unit LSTM.

Tabel 4.19 Hasil pengujian terhadap 2000 unit LSTM pada fitur irisan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	2736	17660	2154	1808	83.73429%	60.21127%
2	1	0.5	594	19530	284	3950	82.61762%	13.07218%
3	2	0.1	1297	19063	751	3247	83.58650%	28.54313%
4	2	0.5	1306	19500	314	3238	**85.41752%**	28.74120%
5	3	0.1	2890	17576	2238	1654	84.02176%	**63.30035%**
6	3	0.5	1260	19515	299	3284	85.29025%	27.72887%
7	10	0.1	1676	18758	1056	2868	83.89030%	36.88380%
8	10	0.5	1246	19521	293	3298	85.25741%	27.42077%

4.4.1.10 Pengujian 4000 Unit LSTM

Pengujian berikut menggunakan 4000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 4000 unit LSTM.

Tabel 4.20 Hasil pengujian terhadap 4000 unit LSTM pada fitur irisan literatur

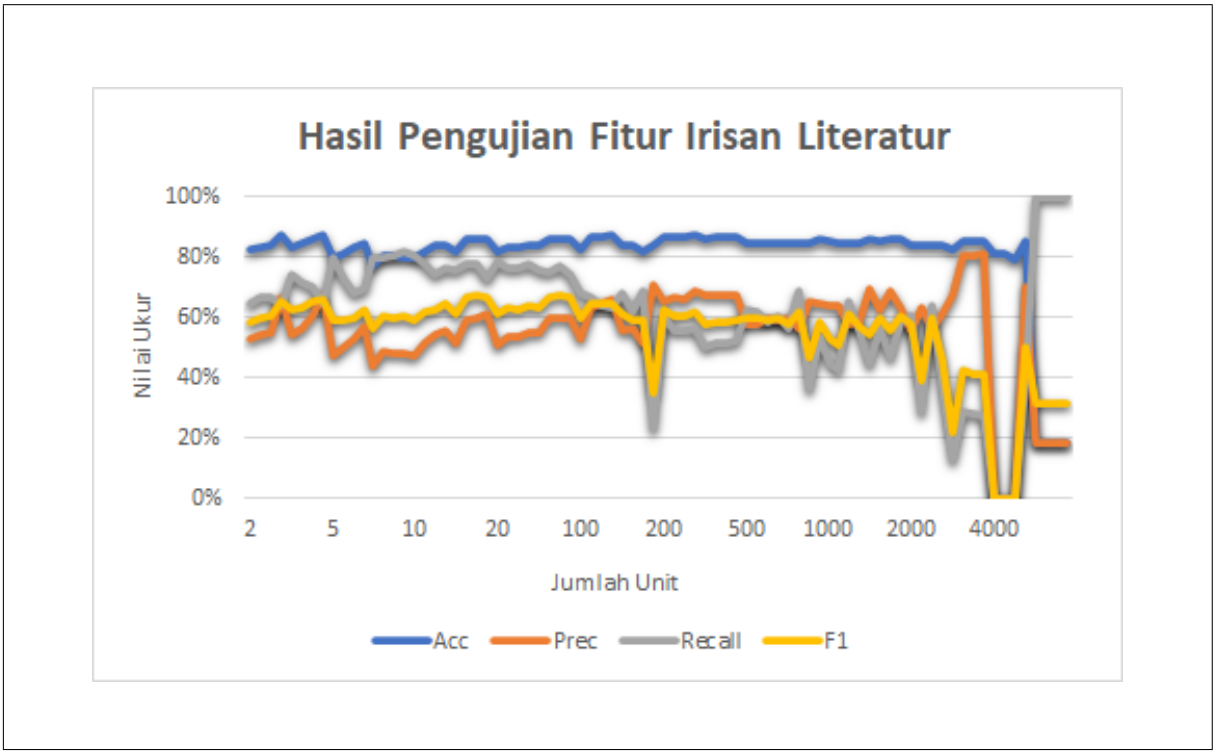
No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	0	18182	0	4176	81.32212%	0%
2	1	0.5	4176	0	18182	0	18.67787%	100%
3	2	0.1	0	18182	0	4176	81.32212%	0%
4	2	0.5	4176	0	18182	0	18.67787%	100%
5	3	0.1	0	18182	0	4176	81.32212%	0%
6	3	0.5	4176	0	18182	0	18.67787%	100%
7	10	0.1	1620	17501	681	2556	**85.52196%**	38.79310%
8	10	0.5	4176	0	18182	0	18.67787%	**100%**

Berikut merupakan tabel hasil rangkuman pengujian fitur irisan literatur.

Tabel 4.21 Rangkuman Hasil Pengujian Fitur Irisan Literatur

No	Jumlah Unit	Epoch	Learning Rate	Akurasi	Precision	Recall	F-Score
1	2	10	0.5	**87.66125%**	67.41812%	64.82618%	66.09675%
2	5	10	0.1	84.60137%	56.97518%	69.48875%	62.61286%
3	10	10	0.5	86.31015%	60.98271%	72.84254%	66.38710%
4	20	3	0.5	86.12651%	59.75320%	**77.28668%**	**67.39829%**
5	100	10	0.1	87.05537%	65.66560%	63.30595%	64.46419%
6	200	10	0.1	87.09381%	68.65220%	56.03626%	61.70599%
7	500	10	0.5	85.89991%	64.45058%	53.54314%	58.49271%
8	1000	3	0.5	86.11089%	68.77160%	46.40661%	55.41772%
9	2000	2	0.5	85.41752%	**80.61728%**	28.74120%	42.37508%
10	4000	10	0.1	85.52196%	70.40417%	38.79310%	50.02316%

Berikut merupakan grafik hasil pengujian fitur irisan literatur.



Gambar 4.3 *Grafik Hasil Pengujian Fitur Irisan Literatur*

Berdasarkan Tabel 4.21, hasil pengujian terhadap fitur irisan literatur mencapai akurasi tertinggi 87.66125% dengan 2 jumlah unit, 10 *epoch*, dan *learning rate* 0.5. *Recall* tertinggi adalah 77.28668% dengan parameter 20 jumlah unit, 3 *epoch*, dan *learning rate* 0.5. *F-Score* tertinggi adalah 67.39829% dengan parameter 20 jumlah unit, 3 *epoch*, dan *learning rate* 0.5 yaitu parameter yang sama untuk mencapai nilai *recall* tertinggi.

Berdasarkan grafik pada Gambar 4.3, hasil pengujian fitur irisan literatur menunjukkan hasil yang tidak stabil. Nilai rata-rata akurasi dari pengujian fitur irisan literatur adalah 80.96645% dan nilai rata-rata *recall* adalah 61.12516%. Kegagalan hasil pengujian fitur irisan literatur terjadi pada jumlah unit 4000, yang ditunjukkan dengan ketidakmampuan sistem mendeteksi label dengan benar pada Tabel 4.20.

4.4.2 Pengujian Fitur Peneliti

Berikut merupakan hasil pengujian terhadap fitur peneliti.

4.4.2.1 Pengujian 2 Unit LSTM

Unit LSTM terkecil untuk dapat dilakukan sebuah iterasi adalah 2. Berikut hasil pengujian untuk 2 unit LSTM.

Tabel 4.22 Hasil pengujian terhadap 2 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	946	21306	160	3944	84.42859%	19.34560%
2	1	0.5	932	21286	180	3958	84.29959%	19.05930%
3	2	0.1	944	21310	156	3946	84.43618%	19.30470%
4	2	0.5	945	21286	180	3945	84.34891%	19.32515%
5	3	0.1	944	21308	158	3946	84.42859%	19.30470%
6	3	0.5	946	21295	171	3944	84.38685%	**19.34560%**
7	10	0.1	944	21305	161	3946	84.41721%	19.05930%
8	10	0.5	933	21325	141	3957	**90.52587%**	19.07975%

4.4.2.2 Pengujian 5 Unit LSTM

Pengujian berikut menggunakan 5 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 5 unit LSTM.

Tabel 4.23 Hasil pengujian terhadap 5 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	828	21355	108	4062	84.17637%	16.93252%
2	1	0.5	932	21312	151	3958	**84.40784%**	19.05930%
3	2	0.1	797	21381	82	4093	84.15740%	16.29857%
4	2	0.5	920	21312	151	3970	84.36231%	18.81391%
5	3	0.1	798	21397	66	4092	84.22191%	16.31902%
6	3	0.5	919	21313	150	3971	84.36231%	18.79346%
7	10	0.1	794	21403	60	4096	84.22949%	16.23722%
8	10	0.5	903	21339	124	3987	84.40025%	**31.24567%**

4.4.2.3 Pengujian 10 Unit LSTM

Pengujian berikut menggunakan 10 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 10 unit LSTM.

Tabel 4.24 Hasil pengujian terhadap 10 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	732	21401	57	4158	84.00258%	14.96933%
2	1	0.5	750	21372	86	4140	83.96083%	15.33742%
3	2	0.1	785	21403	55	4105	84.21132%	15.33742%
4	2	0.5	920	21322	136	3970	**84.41627%**	18.81391%
5	3	0.1	766	21406	52	4124	84.15059%	15.66462%
6	3	0.5	930	21311	147	3960	84.41247%	**19.01840%**
7	10	0.1	736	21408	50	4154	84.04432%	15.05112%
8	10	0.5	900	21339	119	3990	84.40488%	18.40491%

4.4.2.4 Pengujian 20 Unit LSTM

Pengujian berikut menggunakan 20 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 20 unit LSTM.

Tabel 4.25 Hasil pengujian terhadap 20 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	717	21404	47	4170	83.988911	14.67158%
2	1	0.5	668	21397	54	4219	83.77629%	13.66892%
3	2	0.1	728	21404	47	4159	84.03067%	14.89666%
4	2	0.5	666	21396	55	4221	83.76490%	13.62799%
5	3	0.1	730	21406	45	4157	84.04586%	14.93759%
6	3	0.5	666	21397	54	4221	83.76869%	13.62799%
7	10	0.1	957	21165	286	3930	83.99271%	**19.58257%**
8	10	0.5	898	21335	116	3989	**84.41415%**	18.37528%

4.4.2.5 Pengujian 100 Unit LSTM

Pengujian berikut menggunakan 100 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 100 unit LSTM.

Tabel 4.26 Hasil pengujian terhadap 100 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	764	21185	203	4106	83.58976%	**15.68789%**
2	1	0.5	646	21302	86	4224	83.58595%	13.26489%
3	2	0.1	724	21342	46	4146	89.04334%	14.86653%
4	2	0.5	676	21248	140	4194	83.49455%	13.88090%
5	3	0.1	727	21339	49	4143	**90.22012%**	14.92813%
6	3	0.5	682	21312	76	4188	83.76113%	14.00411%
7	10	0.1	746	21324	64	4124	84.05057%	15.31828%
8	10	0.5	716	21312	76	4154	83.89062%	14.70226%

4.4.2.6 Pengujian 200 Unit LSTM

Pengujian berikut menggunakan 200 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 200 unit LSTM.

Tabel 4.27 Hasil pengujian terhadap 200 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	696	21248	56	4158	**90.56121%**	14.33869%
2	1	0.5	661	21226	78	4193	83.67229%	13.61763%
3	2	0.1	706	21240	64	4148	83.89785%	14.54471%
4	2	0.5	651	21249	55	4203	83.72199%	13.41162%
5	3	0.1	1061	20957	347	3793	84.17310%	21.85826%
6	3	0.5	653	21249	55	4201	83.72964%	13.45282%
7	10	0.1	1858	21058	246	2996	85.12500%	**38.27771%**
8	10	0.5	655	21248	56	4199	83.73346%	13.49403%

4.4.2.7 Pengujian 500 Unit LSTM

Pengujian berikut menggunakan 500 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 500 unit LSTM.

Tabel 4.28 Hasil pengujian terhadap 500 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	740	21003	57	4058	84.08616%	15.42309%
2	1	0.5	4798	0	21060	0	18.55518%	100%
3	2	0.1	958	20930	130	3840	84.64691%	19.96665%
4	2	0.5	4798	0	21060	0	18.55518%	100%
5	3	0.1	1912	20865	195	2886	88.08492%	39.84994%
6	3	0.5	4798	0	21060	0	18.55518%	**100%**
7	10	0.1	2299	20876	184	2499	**89.62410%**	**47.91580%**
8	10	0.5	881	20911	149	3917	84.27565%	18.36182%

4.4.2.8 Pengujian 1000 Unit LSTM

Pengujian berikut menggunakan 1000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 1000 unit LSTM.

Tabel 4.29 Hasil pengujian terhadap 1000 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	1612	20141	500	3105	85.78357%	34.17426%
2	1	0.5	4717	0	20641	0	18.60162%	100%
3	2	0.1	1634	20146	495	3083	85.89005%	34.64066%
4	2	0.5	4717	0	20641	0	18.60162%	100%
5	3	0.1	3601	19370	1271	1116	**90.58680%**	**76.34089%**
6	3	0.5	4717	0	20641	0	18.60162%	100%
7	10	0.1	1254	20371	270	3463	85.27880%	26.58469%
8	10	0.5	4717	0	20641	0	18.60162%	**100%**

4.4.2.9 Pengujian 2000 Unit LSTM

Pengujian berikut menggunakan 2000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 2000 unit LSTM.

Tabel 4.30 Hasil pengujian terhadap 2000 unit LSTM pada fitur peneliti

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4544	0	19814	0	18.65506%	100%
2	1	0.5	4544	0	19814	0	18.65506%	100%
3	2	0.1	4544	0	19814	0	18.65506%	100%
4	2	0.5	4544	0	19814	0	18.65506%	100%
5	3	0.1	2337	18780	1034	2207	86.69430%	**51.43046%**
6	3	0.5	4544	0	19814	0	18.65506%	100%
7	10	0.1	2155	19610	204	204	**89.35462%**	47.42518%
8	10	0.5	4544	0	19814	0	18.65506%	**100%**

4.4.2.10 Pengujian 4000 Unit LSTM

Pengujian berikut menggunakan 4000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 4000 unit LSTM.

Tabel 4.31 Hasil pengujian terhadap 4000 unit LSTM pada fitur peneliti

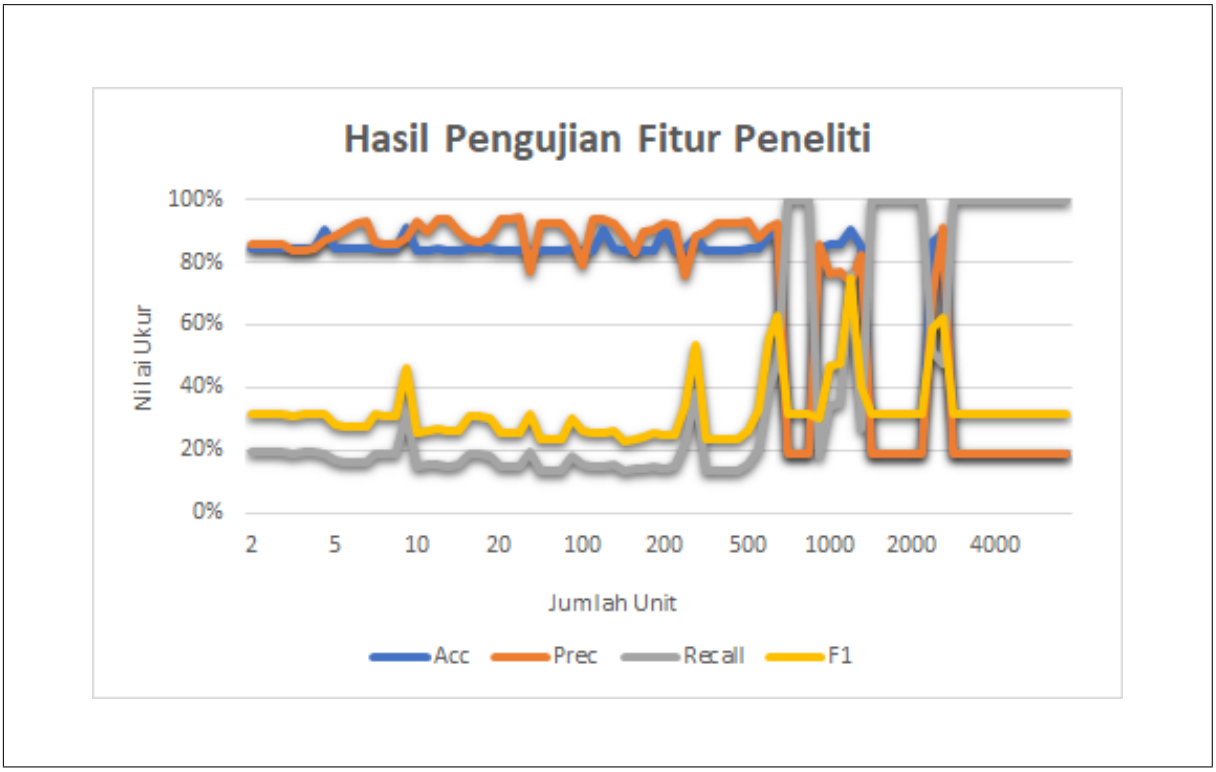
No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4176	0	18182	0	18.67787%	100%
2	1	0.5	4176	0	18182	0	18.67787%	100%
3	2	0.1	4176	0	18182	0	18.67787%	100%
4	2	0.5	4176	0	18182	0	18.67787%	100%
5	3	0.1	4176	0	18182	0	18.67787%	100%
6	3	0.5	4176	0	18182	0	18.67787%	100%
7	10	0.1	4176	0	18182	0	18.67787%	100%
8	10	0.5	4176	0	18182	0	18.67787%	100%

Berikut merupakan tabel rangkuman hasil pengujian fitur peneliti.

Tabel 4.32 Rangkuman Hasil Pengujian Fitur Peneliti

No	Jumlah Unit	Epoch	Learning Rate	Akurasi	Precision	Recall	F-Score
1	2	10	0.5	90.52587%	86.87151%	19.07975%	31.28773%
2	5	1	0.5	84.40784%	86.05725%	19.05930%	31.20710%
3	10	2	0.5	84.41627%	87.12121%	18.81391%	30.94517%
4	20	10	0.5	84.41415%	88.56016%	18.37528%	30.43552%
5	100	3	0.1	90.22012%	**93.68557%**	14.92813%	25.75275%
6	200	1	0.1	**90.56121%**	92.55319%	14.33869%	24.83054%
7	500	10	0.1	89.62410%	92.58961%	47.91580%	63.15067%
8	1000	3	0.1	85.91371%	73.91215%	**76.34089%**	**75.10689%**
9	2000	10	0.1	89.35462%	91.35227%	47.42518%	62.43662%
10	4000	1	0.1	18.67787%	18.67788%	**100%**	31.47660%

Berikut merupakan grafik hasil pengujian terhadap fitur peneliti.



Gambar 4.4 *Grafik Hasil Pengujian Fitur Peneliti*

Hasil pengujian terhadap fitur peneliti berdasarkan Tabel 4.32 mencapai akurasi tertinggi 90.56121% dengan parameter pengujian 200 unit, 1 *epoch* dan *learning rate* 0.1. *Recall* dan *F-Score* tertinggi didapatkan pada pengujian 1000 unit, *epoch* 3 dan *learning rate* 0.1 dengan nilai *Recall* 76.34089% dan *F-Score* 75.10689%. *Recall* sempurna didapatkan yaitu 100% namun dengan nilai akurasi 18.67787% disebabkan oleh kegagalan mendeteksi aktivitas *benign* yang menyebabkan jumlah *true positive* dan *false negative* yang sempurna seperti tercantum pada tTabel 4.31.

Berdasarkan grafik pada Gambar 4.4, dapat dianalisis bahwa hasil pengujian memiliki kenaikan dan penurunan hasil yang drastis. Pada jumlah unit di atas 1000, terjadi penurunan nilai akurasi dan *F-Score*. Nilai *recall* pada pengujian terhadap fitur peneliti tergolong buruk, dengan nilai rata-rata 41.69396%, sedangkan akurasi memiliki nilai rata-rata 67.63889%. *F-Score* dari pengujian fitur peneliti memiliki rata-rata 32.07608%.

4.4.3 Pengujian Fitur Kombinasi

Berikut merupakan hasil pengujian terhadap fitur kombinasi.

4.4.3.1 Pengujian 2 Unit LSTM

Unit LSTM terkecil untuk dapat dilakukan sebuah iterasi adalah 2. Berikut hasil pengujian untuk 2 unit LSTM.

Tabel 4.33 Hasil pengujian terhadap 2 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3538	20171	1295	1352	89.95674%	72.35174%
2	1	0.5	3610	20016	1450	1280	89.64183%	**73.82413%**
3	2	0.1	3575	20212	1254	1315	90.25269%	73.10838%
4	2	0.5	3582	20167	1299	1308	90.10851%	73.25153%
5	3	0.1	3541	20267	1199	1349	90.33237%	72.92010%
6	3	0.5	3540	20245	1221	1350	90.24511%	72.39264%
7	10	0.1	3479	20395	1071	1411	**90.58279%**	71.14519%
8	10	0.5	3494	20365	1101	1396	90.52587%	71.45194%

4.4.3.2 Pengujian 5 Unit LSTM

Pengujian berikut menggunakan 5 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 5 unit LSTM.

Tabel 4.34 Hasil pengujian terhadap 5 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3595	20183	1280	1295	90.22882%	73.51738%
2	1	0.5	4429	19002	2461	461	88.91208%	90.57260%
3	2	0.1	4046	18976	2487	844	87.36001%	82.74029%
4	2	0.5	4461	19078	2385	429	89.32189%	**91.22699%**
5	3	0.1	4090	19422	2041	800	89.21944%	83.64008%
6	3	0.5	4443	19133	2330	447	89.46230%	90.85890%
7	10	0.1	4034	19801	1662	856	**90.44511%**	82.49489%
8	10	0.5	4460	19236	2227	430	89.91766%	91.20654%

4.4.3.3 Pengujian 10 Unit LSTM

Pengujian berikut menggunakan 10 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 10 unit LSTM.

Tabel 4.35 Hasil pengujian terhadap 10 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3630	20103	1355	1260	90.07515%	74.23313%
2	1	0.5	3971	19624	1834	919	89.55139%	81.20654%
3	2	0.1	3973	19590	1868	917	89.42994%	81.24744%
4	2	0.5	3949	19899	1559	941	90.51161%	80.75665%
5	3	0.1	3835	19861	1597	1055	89.93472%	78.42536%
6	3	0.5	3983	19952	1506	907	90.84110%	81.45194%
7	10	0.1	3749	20010	1448	1141	90.17383%	76.66667%
8	10	0.5	4027	20041	1417	863	**91.34659%**	**82.35174%**

4.4.3.4 Pengujian 20 Unit LSTM

Pengujian berikut menggunakan 20 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 20 unit LSTM.

Tabel 4.36 Hasil pengujian terhadap 20 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4170	19040	2411	717	88.12362%	85.32842%
2	1	0.5	3915	19759	1692	972	89.88534%	80.11050%
3	2	0.1	4106	19476	1975	781	89.53603%	84.01883%
4	2	0.5	3933	19901	1550	954	90.49282%	80.47882%
5	3	0.1	4148	19619	1832	739	90.23844%	84.87825%
6	3	0.5	3877	20027	1424	1010	90.78599%	79.33292%
7	10	0.1	4353	19421	2030	534	90.26502%	**89.07305%**
8	10	0.5	3957	20042	1409	930	**91.11929%**	80.96992%

4.4.3.5 Pengujian 100 Unit LSTM

Pengujian berikut menggunakan 100 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 100 unit LSTM.

Tabel 4.37 Hasil pengujian terhadap 100 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4187	19407	1981	683	89.85452%	85.97536%
2	1	0.5	3460	20060	1328	1410	89.57270%	71.04723%
3	2	0.1	3714	19667	1721	1156	89.04334%	76.26283%
4	2	0.5	3986	19644	1744	974	89.64887%	80.36290%
5	3	0.1	4124	19566	1822	746	90.22012%	84.68172%
6	3	0.5	3931	19867	1521	939	90.63143%	80.71869%
7	10	0.1	4277	19566	1822	593	90.80280%	**87.82341%**
8	10	0.5	3972	20214	1174	898	**92.10907%**	81.56057%

4.4.3.6 Pengujian 200 Unit LSTM

Pengujian berikut menggunakan 200 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 200 unit LSTM.

Tabel 4.38 Hasil pengujian terhadap 200 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3492	20197	1107	1362	90.56121%	71.94607%
2	1	0.5	3883	19886	1418	971	90.86704%	79.99588%
3	2	0.1	3674	19954	1350	1180	90.32801%	75.69015%
4	2	0.5	3556	20349	955	1298	91.38696%	73.25917%
5	3	0.1	3806	19864	1440	1048	90.48857%	78.40956%
6	3	0.5	3777	20350	954	1077	92.23564%	77.81211%
7	10	0.1	3886	20368	936	968	92.72115%	80.05768%
8	10	0.5	4149	20236	1068	705	**93.22196%**	**85.47590%**

4.4.3.7 Pengujian 500 Unit LSTM

Pengujian berikut menggunakan 500 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 500 unit LSTM.

Tabel 4.39 Hasil pengujian terhadap 500 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3652	20059	1001	1146	91.69696%	76.11505%
2	1	0.5	3916	19841	1219	882	91.87485%	81.61734%
3	2	0.1	3774	20077	983	1024	92.23838%	78.65777%
4	2	0.5	4142	19749	1311	656	92.39307%	86.32764%
5	3	0.1	3772	20203	857	1026	92.71792%	78.61609%
6	3	0.5	4165	19880	1180	633	92.98863%	86.80700%
7	10	0.1	3880	20156	904	918	92.95382%	80.86703%
8	10	0.5	4201	19952	1108	597	**93.40629%**	**87.55732%**

4.4.3.8 Pengujian 1000 Unit LSTM

Pengujian berikut menggunakan 1000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 1000 unit LSTM.

Tabel 4.40 Hasil pengujian terhadap 1000 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3394	19891	750	1323	91.82507%	71.95251%
2	1	0.5	3601	19370	1271	1116	90.58679%	76.34089%
3	2	0.1	3544	19990	651	1173	92.80700%	75.13250%
4	2	0.5	3474	19952	689	1243	92.38110%	73.64851%
5	3	0.1	3634	20041	600	1083	93.36304%	77.04049%
6	3	0.5	3598	19904	737	1119	92.68081%	76.27729%
7	10	0.1	3652	20114	527	1065	**93.72190%**	**77.42209%**
8	10	0.5	3594	20140	501	1123	93.59571%	76.19250%

4.4.3.9 Pengujian 2000 Unit LSTM

Pengujian berikut menggunakan 2000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 2000 unit LSTM.

Tabel 4.41 Hasil pengujian terhadap 2000 unit LSTM pada fitur kombinasi

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3130	19152	662	1414	91.47713%	68.88204%
2	1	0.5	3715	18559	1255	829	91.44429%	81.75616%
3	2	0.1	3584	18886	928	960	92.24895%	78.87324%
4	2	0.5	3763	18779	1035	781	92.54454%	**82.81250%**
5	3	0.1	3452	19273	541	1092	93.29584%	75.96831%
6	3	0.5	3162	19291	523	1382	92.17916%	69.58627%
7	10	0.1	3550	19252	562	994	93.61196%	78.12500%
8	10	0.5	3651	19213	601	893	**93.86649%**	80.34771%

4.4.3.10 Pengujian 4000 Unit LSTM

Pengujian berikut menggunakan 4000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 4000 unit LSTM.

Tabel 4.42 Hasil pengujian terhadap 4000 unit LSTM pada fitur kombinasi

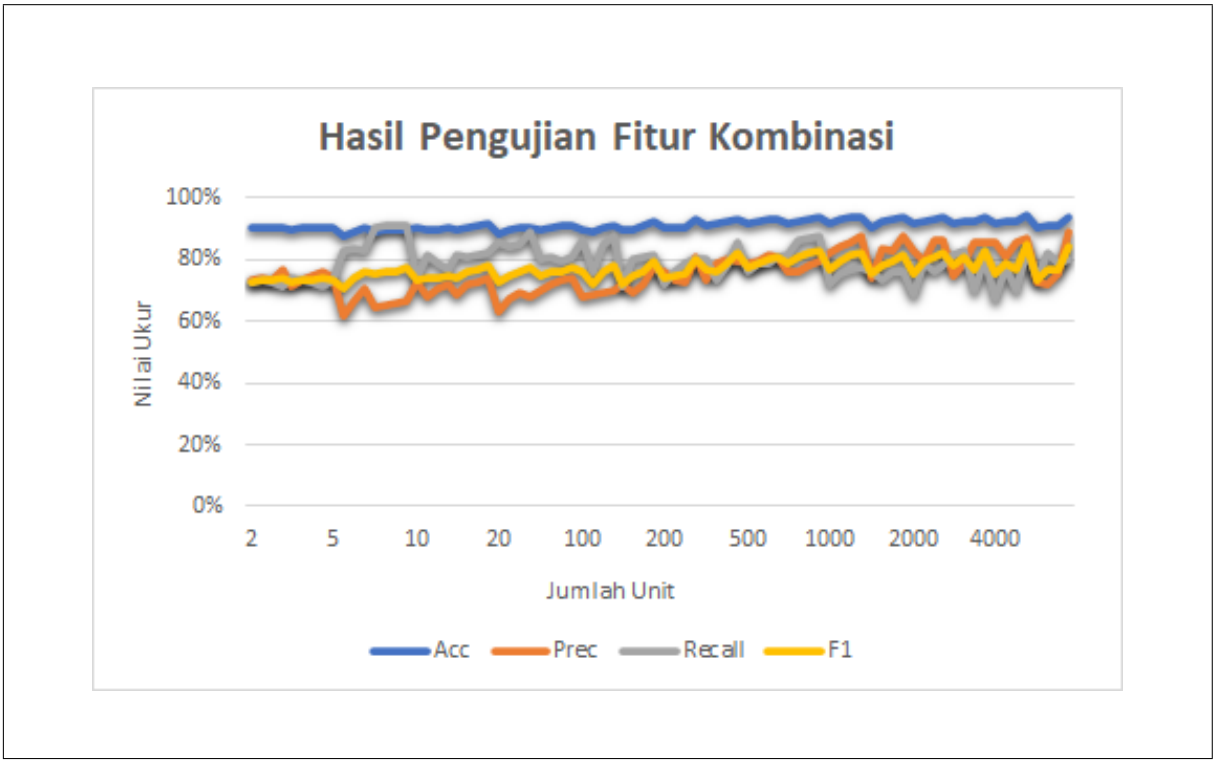
No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	2796	17715	467	1380	91.73897%	66.95402%
2	1	0.5	3089	17025	1157	1087	89.96332%	73.97031%
3	2	0.1	3218	17428	754	958	92.34279%	77.05939%
4	2	0.5	3444	16853	1329	732	90.78182%	**82.47126%**
5	3	0.1	2915	17700	482	1261	93.20413%	69.80634%
6	3	0.5	3288	17104	1078	888	91.20673%	78.73563%
7	10	0.1	3459	17662	520	717	**94.46730%**	82.83046%
8	10	0.5	3913	17691	491	983	93.40783%	79.92239%

Berikut tabel rangkuman hasil pengujian terhadap fitur kombinasi.

Tabel 4.43 Rangkuman Hasil Pengujian Fitur Kombinasi

No	Jumlah Unit	Epoch	Learning Rate	Akurasi	Precision	Recall	F-Score
1	2	10	0.1	90.58279%	76.46154%	71.14519%	73.70763%
2	5	10	0.1	90.44511%	70.82163%	82.49489%	76.21387%
3	10	10	0.5	91.34659%	73.97134%	82.35174%	77.93691%
4	20	10	0.5	91.11929%	73.74208%	80.96992%	77.18716%
5	100	10	0.5	92.10907%	77.18616%	81.56057%	79.31310%
6	200	10	0.5	93.22196%	79.52846%	85.47590%	82.39500%
7	500	10	0.5	93.40629%	79.12978%	**87.55732%**	83.13050%
8	1000	10	0.1	93.72190%	**87.38933%**	77.42209%	82.10432%
9	2000	10	0.5	93.86649%	85.86548%	80.34771%	83.01501%
10	4000	10	0.1	**94.46730%**	86.93139%	82.83046%	**84.83139%**

Berikut merupakan grafik hasil pengujian terhadap fitur kombinasi.



Gambar 4.5 *Grafik Hasil Pengujian Fitur Gabungan*

Hasil pengujian terhadap fitur kombinasi mencapai akurasi tertinggi 94.46730% dengan jumlah unit 4000, 10 *epoch*, dan *learning rate* 0.1. *Recall* tertinggi yang berhasil dcapai adalah adalah 87.55732% dengan parameter 500 jumlah unit, 10 *epoch*, dan *learning rate* 0.5. *F-Score* tertinggi adalah 84.83139% dengan parameter 4000 jumlah unit, 10 *epoch*, dan *learning rate* 0.1 yaitu parameter yang sama untuk mencapai nilai akurasi tertinggi.

Berdasarkan grafik pada Gambar 4.5, hasil pengujian fitur kombinasi menunjukkan hasil yang stabil meningkat. Nilai rata-rata akurasi dari pengujian fitur kombinasi adalah 91.13148% dan nilai rata-rata *recall* adalah 79.08722%. Tidak seperti pengujian terhadap fitur lain, fitur kombinasi memiliki hasil pengujian yang stabil terhadap pelbagai variasi parameter.

4.4.4 Pengujian Fitur Gabungan Literatur

Berikut merupakan hasil pengujian terhadap fitur gabungan literatur.

4.4.4.1 Pengujian 2 Unit LSTM

Unit LSTM terkecil untuk dapat dilakukan sebuah iterasi adalah 2. Berikut hasil pengujian untuk 2 unit LSTM.

Tabel 4.44 Hasil pengujian terhadap 2 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3615	20361	1107	1277	90.95599%	73.89616%
2	1	0.01	3247	20051	1417	1645	88.38392%	66.37367%
3	2	0.1	3916	20439	1029	976	92.39378%	80.04906%
4	2	0.01	3402	20079	1389	1490	89.07815%	69.54211%
5	3	0.1	4025	20410	1058	867	**92.69727%**	82.27719%
6	3	0.01	3509	20107	1361	1383	89.59029%	71.72935%
7	10	0.1	4109	20274	1194	783	92.50000%	**83.99428%**
8	10	0.01	3584	20324	1144	1308	90.69803%	73.26247%

4.4.4.2 Pengujian 5 Unit LSTM

Pengujian berikut menggunakan 5 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 5 unit LSTM.

Tabel 4.45 Hasil pengujian terhadap 5 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3905	20250	1216	986	91.64548%	79.84052%
2	1	0.01	3219	20627	839	1672	90.47312%	65.81476%
3	2	0.1	3946	20421	1045	945	92.14982%	80.67880%
4	2	0.01	3298	20664	802	1593	90.91323%	67.42997%
5	3	0.1	4581	19630	1836	310	91.85795%	93.66183%
6	3	0.01	3342	20673	793	1549	91.11431%	68.32958%
7	10	0.1	4615	19723	1743	276	**92.33980%**	**94.35698%**
8	10	0.01	3748	20638	828	1143	92.12191%	76.63055%

4.4.4.3 Pengujian 10 Unit LSTM

Pengujian berikut menggunakan 10 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 10 unit LSTM.

Tabel 4.46 Hasil pengujian terhadap 10 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4533	19535	1926	358	91.33273%	92.68043%
2	1	0.01	3445	20610	851	1446	91.28339%	70.43549%
3	2	0.1	4616	19565	1896	275	91.76154%	94.37743%
4	2	0.01	3458	20667	794	1433	91.54903%	70.70129%
5	3	0.1	4658	19537	1924	233	91.81466%	95.23615%
6	3	0.01	3543	20673	788	1348	91.89435%	72.43917%
7	10	0.1	4676	19487	1974	215	91.69323%	**95.60417%**
8	10	0.01	4005	20622	839	886	**93.45401%**	81.88510%

4.4.4.4 Pengujian 20 Unit LSTM

Pengujian berikut menggunakan 20 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 20 unit LSTM.

Tabel 4.47 Hasil pengujian terhadap 20 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3779	20323	1131	1109	91.49647%	77.31178%
2	1	0.01	3510	20655	799	1378	91.73563%	71.80851%
3	2	0.1	3948	20515	939	940	92.86690%	80.76923%
4	2	0.01	3576	20684	770	1312	92.09627%	73.15876%
5	3	0.1	3855	20601	853	1033	92.84033%	78.86661%
6	3	0.01	3620	20698	756	1268	92.31645%	74.05892%
7	10	0.1	3996	20713	741	892	**93.80077%**	**81.75123%**
8	10	0.01	3949	20683	771	939	93.50847%	80.78969%

4.4.4.5 Pengujian 100 Unit LSTM

Pengujian berikut menggunakan 100 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 100 unit LSTM.

Tabel 4.48 Hasil pengujian terhadap 100 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3714	20542	849	1157	92.36159%	76.24718%
2	1	0.01	3564	20623	768	1307	92.09885%	73.16773%
3	2	0.1	3762	20591	800	1109	92.73094%	77.23260%
4	2	0.01	3814	20628	763	1057	93.06983%	78.30014%
5	3	0.1	3747	20575	816	1124	92.61290%	76.92466%
6	3	0.01	3899	20642	749	972	93.44681%	80.04517%
7	10	0.1	3625	20584	807	1246	92.18262%	74.42004%
8	10	0.01	3957	20689	702	914	**93.84662%**	**81.23589%**

4.4.4.6 Pengujian 200 Unit LSTM

Pengujian berikut menggunakan 200 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 200 unit LSTM.

Tabel 4.49 Hasil pengujian terhadap 200 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3861	20506	800	995	93.13890%	79.50988%
2	1	0.01	3749	20561	745	1107	92.92103%	77.20346%
3	2	0.1	3762	20578	728	1094	93.03570%	77.47117%
4	2	0.01	3829	20601	705	1027	93.37971%	78.85091%
5	3	0.1	3666	20616	690	1190	92.81401%	75.49423%
6	3	0.01	3895	20614	692	961	93.68168%	**80.21005%**
7	10	0.1	3840	20685	621	1016	**93.74283%**	79.07743%
8	10	0.01	3880	20634	672	976	93.70079%	79.90115%

4.4.4.7 Pengujian 500 Unit LSTM

Pengujian berikut menggunakan 500 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 500 unit LSTM.

Tabel 4.50 Hasil pengujian terhadap 500 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3649	20377	685	1151	92.90078%	76.02083%
2	1	0.01	3716	20430	632	1084	93.36478%	77.41667%
3	2	0.1	3826	20285	777	974	93.22945%	79.70833%
4	2	0.01	3711	20446	616	1089	93.40732%	77.31250%
5	3	0.1	3845	20298	764	955	93.35318%	**80.10417%**
6	3	0.01	3725	20445	617	1075	93.45758%	77.60417%
7	10	0.1	3613	20342	720	1187	92.62625%	75.27083%
8	10	0.01	3688	20512	559	1112	**93.54103%**	76.83333%

4.4.4.8 Pengujian 1000 Unit LSTM

Pengujian berikut menggunakan 1000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 1000 unit LSTM.

Tabel 4.51 Hasil pengujian terhadap 1000 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3660	20197	447	1058	94.06593%	77.57524%
2	1	0.01	3775	19919	725	943	93.42323%	**80.01272%**
3	2	0.1	3701	20186	458	1017	**94.18421%**	78.44426%
4	2	0.01	3663	20010	634	1055	93.34043%	77.63883%
5	3	0.1	3654	20221	423	1064	94.13690%	77.44807%
6	3	0.01	3682	20007	637	1036	93.40352%	78.04154%
7	10	0.1	3649	20231	413	1069	94.15661%	77.34209%
8	10	0.01	3728	20037	607	990	93.70318%	79.01653%

4.4.4.9 Pengujian 2000 Unit LSTM

Pengujian berikut menggunakan 2000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 2000 unit LSTM.

Tabel 4.52 Hasil pengujian terhadap 2000 unit LSTM pada fitur gabungan literatur

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3556	18809	1008	989	91.80281%	78.23982%
2	1	0.01	3791	19040	777	754	93.71562%	83.41034%
3	2	0.1	3579	18780	1037	966	91.77818%	78.74587%
4	2	0.01	3812	19054	763	733	93.85929%	83.87239%
5	3	0.1	3034	19109	708	1511	90.89155%	66.75468%
6	3	0.01	3863	19026	791	682	**93.95370%**	84.99450%
7	10	0.1	3613	18787	1030	932	91.94647%	79.49395%
8	10	0.01	3884	18994	823	661	93.90855%	**85.45655%**

4.4.4.10 Pengujian 4000 Unit LSTM

Pengujian berikut menggunakan 4000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 4000 unit LSTM.

Tabel 4.53 Hasil pengujian terhadap 4000 unit LSTM pada fitur gabungan literatur

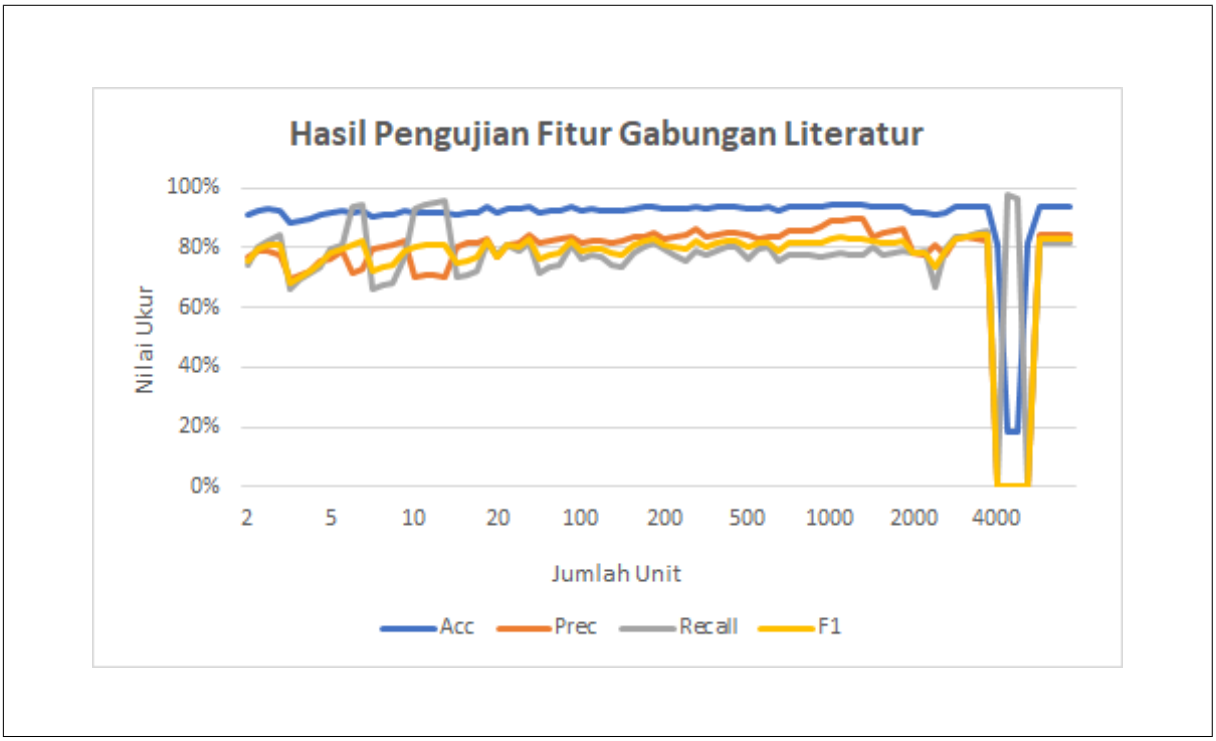
No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	71	18072	113	4106	81.13317%	1.69978%
2	1	0.01	3401	17561	624	776	93.73938%	81.42207%
3	2	0.1	4078	92	18093	99	18.64771%	**97.62988%**
4	2	0.01	3399	17564	621	778	93.74385%	81.37419%
5	3	0.1	4024	131	18054	153	18.58063%	96.33708%
6	3	0.01	3396	17563	622	781	93.72596%	81.30237%
7	10	0.1	0	18185	0	4177	81.32099%	0%
8	10	0.01	3411	17561	624	766	**93.78410%**	81.66148%

Berikut tabel rangkuman hasil pengujian fitur gabungan literatur.

Tabel 4.54 Rangkuman Hasil Pengujian Fitur Gabungan Literatur

No	Jumlah Unit	Epoch	Learning Rate	Akurasi	Precision	Recall	F-Score
1	2	3	0.1	92.69727%	79.18552%	82.27719%	80.70175%
2	5	10	0.1	92.33980%	72.58572%	**94.35698%**	82.05174%
3	10	10	0.1	93.45401%	82.67960%	81.88510%	82.28043%
4	20	10	0.1	93.80077%	84.35719%	81.75123%	83.03377%
5	100	10	0.01	93.84662%	84.93239%	81.23589%	83.04302%
6	200	10	0.01	93.74283%	86.07935%	79.07743%	82.42997%
7	500	10	0.01	93.54103%	86.83777%	76.83333%	81.52979%
8	1000	2	0.1	**94.18421%**	**88.98774%**	78.44426%	83.38403%
9	2000	3	0.01	93.95370%	83.00387%	84.99450%	**83.98739%**
10	4000	10	0.01	93.78410%	84.53532%	81.66148%	83.07355%

Berikut merupakan grafik hasil pengujian terhadap fitur gabungan literatur.



Gambar 4.6 *Grafik Hasil Pengujian Fitur Gabungan Literatur*

Hasil pengujian terhadap fitur gabungan literatur seperti pada Tabel 4.54 mencapai

akurasi tertinggi 94.18421% dengan jumlah unit 1000, 2 *epoch*, dan *learning rate* 0.1. *Recall* tertinggi yang berhasil dcapai adalah adalah 94.35698% dengan parameter 5 jumlah unit, 10 *epoch*, dan *learning rate* 0.5. *F-Score* tertinggi adalah 83.98739% dengan parameter 2000 jumlah unit, 3 *epoch*, dan *learning rate* 0.01.

Berdasarkan grafik pada Gambar 4.6, hasil pengujian fitur gabungan literatur menunjukkan hasil yang stabil meningkat kecuali pada pengujian terhadap unit 4000. Nilai rata-rata akurasi dari pengujian fitur gabungan literatur adalah 90.45810% dan nilai rata-rata *recall* adalah 77.16523%. Pengujian sistem terhadap fitur gabungan literatur memiliki kegagalan pada pengujian 4000 unit LSTM terhadap beberapa parameter yang ditunjukkan oleh Tabel 4.53.

4.4.5 Pengujian Fitur PCA

Berikut merupakan hasil pengujian terhadap fitur PCA.

4.4.5.1 Pengujian 2 Unit LSTM

Unit LSTM terkecil untuk dapat dilakukan sebuah iterasi adalah 2. Berikut hasil pengujian untuk 2 unit LSTM.

Tabel 4.55 Hasil pengujian terhadap 2 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3330	20609	859	1562	90.81563%	68.07032%
2	1	0.01	709	21309	159	4183	83.52807%	14.49305%
3	2	0.1	3609	20772	696	1283	92.49241%	73.77351%
4	2	0.01	1410	20828	640	3482	84.36267%	28.82257%
5	3	0.1	3754	20779	689	1138	**93.06904%**	**76.73753%**
6	3	0.01	1893	20676	792	2999	85.61836%	38.69583%

4.4.5.2 Pengujian 5 Unit LSTM

Pengujian berikut menggunakan 5 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 5 unit LSTM.

Tabel 4.56 Hasil pengujian terhadap 5 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3958	20364	1102	933	92.27909%	80.92415%
2	1	0.01	1585	20718	748	3306	84.61888%	32.40646%
3	2	0.1	4020	20294	1172	871	92.24873%	82.19178%
4	2	0.01	2305	20623	843	2586	86.99017%	47.12738%
5	3	0.1	4112	20259	1207	779	**92.46499%**	**84.07279%**
6	3	0.01	2753	20616	850	2138	88.66335%	56.28706%

4.4.5.3 Pengujian 10 Unit LSTM

Pengujian berikut menggunakan 10 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 10 unit LSTM.

Tabel 4.57 Hasil pengujian terhadap 10 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4179	19964	1467	712	91.73117%	85.44265%
2	1	0.01	2206	20647	814	2685	86.72207%	45.10325%
3	2	0.1	4249	20135	1326	642	92.53187%	**86.87385%**
4	2	0.01	2977	20824	637	1914	90.31952%	60.86690%
5	3	0.1	4247	20228	1233	644	**92.8772%**	86.83296%
6	3	0.01	3156	20929	532	1735	91.39724%	64.52668%

4.4.5.4 Pengujian 20 Unit LSTM

Pengujian berikut menggunakan 20 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 20 unit LSTM.

Tabel 4.58 Hasil pengujian terhadap 20 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	4320	19973	1481	568	92.22154%	**88.37971%**
2	1	0.01	3053	20788	666	1835	90.50565.%	62.45908%
3	2	0.1	4272	20117	1337	616	92.58598%	87.39771%
4	2	0.01	3323	20911	543	1565	91.99757%	67.98282%
5	3	0.1	4269	20173	1281	619	**92.78718%**	87.33633%
6	3	0.01	3456	20894	560	1432	92.43793%	70.70376%

4.4.5.5 Pengujian 100 Unit LSTM

Pengujian berikut menggunakan 100 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 100 unit LSTM.

Tabel 4.59 Hasil pengujian terhadap 100 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3996	20500	891	875	93.27545%	82.03654%
2	1	0.01	3230	20909	482	1641	91.91607.%	66.31082%
3	2	0.1	4065	20497	894	806	93.52678%	**83.45309%**
4	2	0.01	3604	20900	491	1267	93.30591%	73.98891%
5	3	0.1	4060	20497	894	811	93.50772%	83.35044%
6	3	0.01	3796	20904	487	1075	**94.05224%**	77.93061%

4.4.5.6 Pengujian 200 Unit LSTM

Pengujian berikut menggunakan 200 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 200 unit LSTM.

Tabel 4.60 Hasil pengujian terhadap 200 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	3819	20678	628	1037	93.6358%	78.64498%
2	1	0.01	3792	20755	551	1064	93.82692.%	78.08896%
3	2	0.1	3819	20720	586	1037	93.79634%	78.64498%
4	2	0.01	3902	20789	517	954	94.37734%	80.35420%
5	3	0.1	3829	20730	576	1027	93.87279%	78.85091%
6	3	0.01	3928	20821	485	928	**94.59903%**	**80.88962%**

4.4.5.7 Pengujian 500 Unit LSTM

Pengujian berikut menggunakan 500 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 500 unit LSTM.

Tabel 4.61 Hasil pengujian terhadap 500 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	0	21601	1	4800	81.43608%	0%
2	1	0.01	3874	20523	539	926	94.33531.%	80.70833%
3	2	0.1	3683	20622	440	1117	93.97958%	76.72917%
4	2	0.01	3899	20560	502	926	94.33531%	81.22917%
5	3	0.1	3700	20638	424	1100	94.10718%	77.08333%
6	3	0.01	3905	20582	480	895	**94.68331%**	**81.35417%**

4.4.5.8 Pengujian 1000 Unit LSTM

Pengujian berikut menggunakan 1000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 1000 unit LSTM.

Tabel 4.62 Hasil pengujian terhadap 1000 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	0	20604	0	4718	81.39736%	0%
2	1	0.01	3840	20190	454	878	94.74804.%	81.39042%
3	2	0.1	0	20604	0	4718	81.39736%	0%
4	2	0.01	3871	20212	432	847	94.95702%	82.04748%
5	3	0.1	0	20604	0	4718	81.39736%	0%
6	3	0.01	3891	20212	432	827	95.03588%	**82.47139%**
7	500	0.01	3872	20502	142	846	**96.10441%**	82.06867%

4.4.5.9 Pengujian 2000 Unit LSTM

Pengujian berikut menggunakan 2000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 2000 unit LSTM.

Tabel 4.63 Hasil pengujian terhadap 2000 unit LSTM pada fitur PCA

No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	0	19817	0	4515	81.34389%	0%
2	1	0.01	3687	19144	673	858	93.71562.%	81.12211%
3	2	0.1	0	19817	0	4515	81.34389%	0%
4	2	0.01	3732	19136	681	813	93.86749%	82.11221%
5	3	0.1	0	19817	0	4515	81.34389%	0%
6	3	0.01	3753	19160	657	792	**94.05221%**	**82.57426%**

4.4.5.10 Pengujian 4000 Unit LSTM

Pengujian berikut menggunakan 4000 unit LSTM sebagai hidden layer. Berikut hasil pengujian untuk 4000 unit LSTM.

Tabel 4.64 Hasil pengujian terhadap 4000 unit LSTM pada fitur PCA

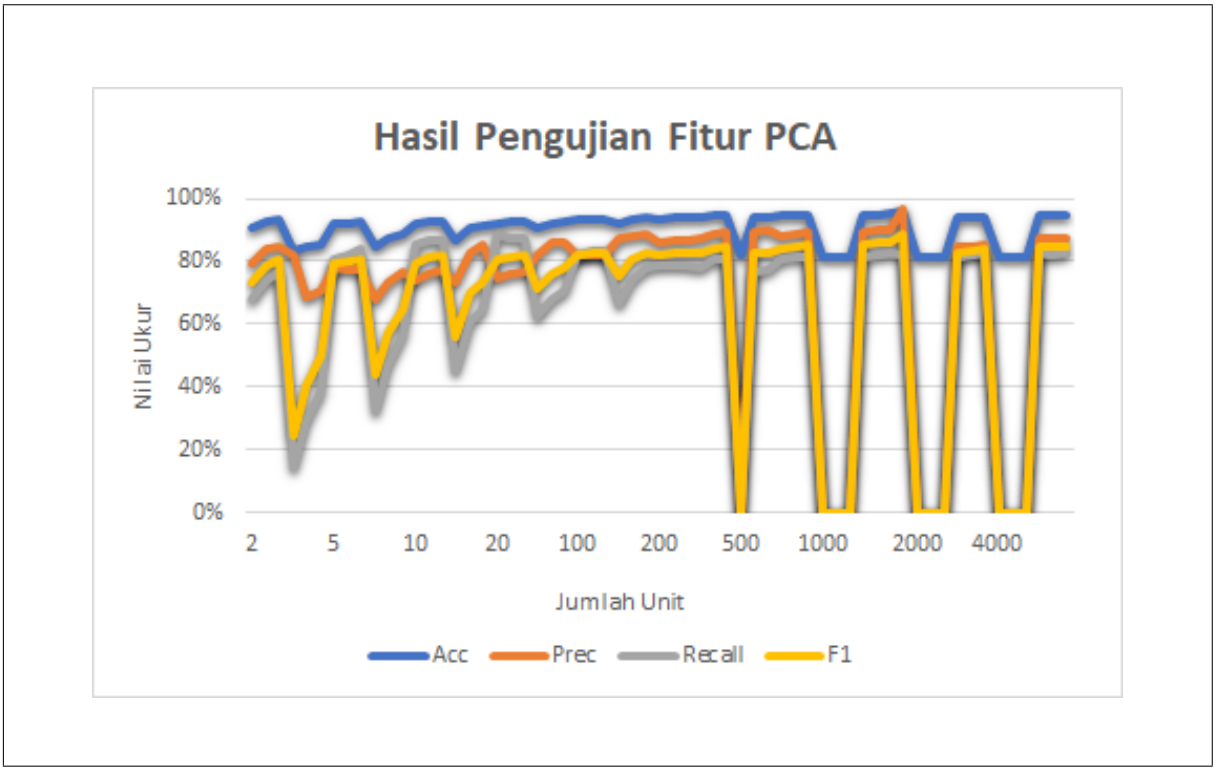
No	Epoch	Learning Rate	True Positive	True Negative	False Positive	False Negative	Akurasi	Recall
1	1	0.1	0	18185	0	4177	81.32099%	0%
2	1	0.01	3444	17679	506	733	94.45935.%	**82.45152%**
3	2	0.1	0	18185	0	4177	81.32099%	0%
4	2	0.01	3433	17683	502	744	94.42804%	82.18817%
5	3	0.1	0	18185	0	4177	81.32099%	0%
6	3	0.01	3442	17690	495	735	**94.49995%**	82.40364%

Berikut tabel rangkuman hasil pengujian fitur PCA

Tabel 4.65 Rangkuman Hasil Pengujian Fitur PCA

No	Jumlah Unit	Epoch	Learning Rate	Akurasi	Precision	Recall	F-Score
1	2	3	0.1	93.06904%	84.49246%	76.73753%	80.42849%
2	5	3	0.1	92.46499%	77.30776%	84.07279%	80.54848%
3	10	3	0.1	92.87720%	77.50000%	86.83296%	81.90146%
4	20	3	0.1	92.78718%	76.91892%	**87.33633%**	81.79728%
5	100	3	0.01	94.05224%	88.62947%	77.93061%	82.93642%
6	200	3	0.01	94.55903%	89.00974%	80.88962%	84.75564%
7	500	3	0.01	94.68331%	89.05359%	81.35417%	85.02994%
8	1000	500	0.01	**96.10441%**	**96.46238%**	82.06867%	**88.68530%**
9	2000	3	0.01	94.05221%	85.10204%	82.57426%	83.81910%
10	4000	3	0.01	94.49995%	87.42697%	82.40364%	84.84102%

Berikut merupakan grafik hasil pengujian terhadap fitur PCA.



Gambar 4.7 *Grafik Hasil Pengujian Fitur PCA*

Hasil pengujian terhadap fitur PCA pada Tabel 4.65 mencapai akurasi tertinggi 96.10441% dengan jumlah unit 1000, 500 *epoch*, dan *learning rate* 0.01. *Recall* tertinggi yang berhasil dicapai adalah adalah 87.33633% dengan parameter 20 jumlah unit, 3 *epoch*, dan *learning rate* 0.1. *F-Score* tertinggi adalah 88.68530% dengan parameter yang sama seperti pengujian dengan akurasi tertinggi yaitu 1000 jumlah unit, 500 *epoch*, dan *learning rate* 0.01.

Berdasarkan grafik pada Gambar 4.7, hasil pengujian fitur PCA menunjukkan hasil yang tidak stabil terhadap variasi jumlah unit. Nilai rata-rata akurasi dari pengujian fitur gabungan literatur adalah 90.50364% dan nilai rata-rata *recall* adalah 61.27846%. Pengujian sistem terhadap fitur gabungan literatur memiliki kegagalan pada pengujian di atas 1000 unit unit LSTM dengan *learning rate* 0.1 yang ditunjukkan oleh Tabel 4.62, Tabel 4.63, dan Tabel 4.64.

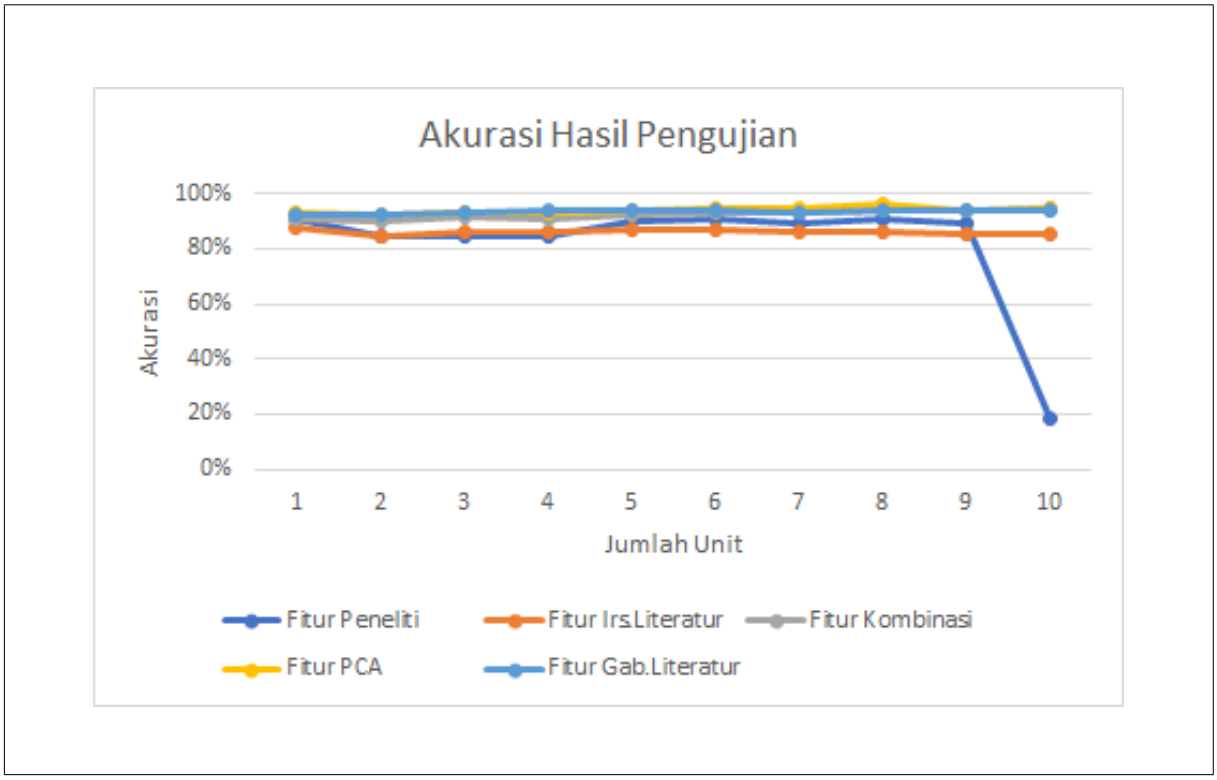
4.5 Analisis Hasil Pengujian

Berdasarkan analisis peneliti terhadap hasil pengujian, parameter yang digunakan sebagai masukan sistem untuk konfigurasi model LSTM memiliki pengaruh signifikan terhadap tingkat akurasi klasifikasi pengujian data. Akurasi tertinggi yang diperoleh dari seluruh rangkaian hasil pengujian adalah 96.10441% dengan konfigurasi 1000 Unit LSTM, 500 *epoch* dan *learning rate* 0.01 terhadap fitur PCA. Sedangkan akurasi terendah dari hasil pengujian adalah 18.67787% dengan konfigurasi 4000 unit LSTM, 10 *epoch* dan *learning rate* 0.5 pada fitur literatur. Berikut tabel hasil pengujian dengan akurasi terbaik terhadap seluruh teknik metode pemilihan fitur set data.

Tabel 4.66 Hasil Pengujian Berdasarkan Akurasi Tertinggi

No	Fitur	Unit	Epoch	LR	Akurasi	Prec	Recall	F-Score
1	Irs.Lit	2	10	0.5	87.66125%	67.41812%	64.82618%	66.09675%
2	Peneliti	1000	3	0.1	90.58680%	73.39122%	76.34090%	75.10690%
3	Kombinasi	4000	10	0.1	94.46730%	86.93139%	82.83046%	84.83139%
4	G.Lit	5	10	0.1	92.33980%	72.58572%	**94.35698%**	82.05174%
5	PCA	1000	500	0.01	**96.10441%**	**96.46238%**	82.06867%	**88.68530%**

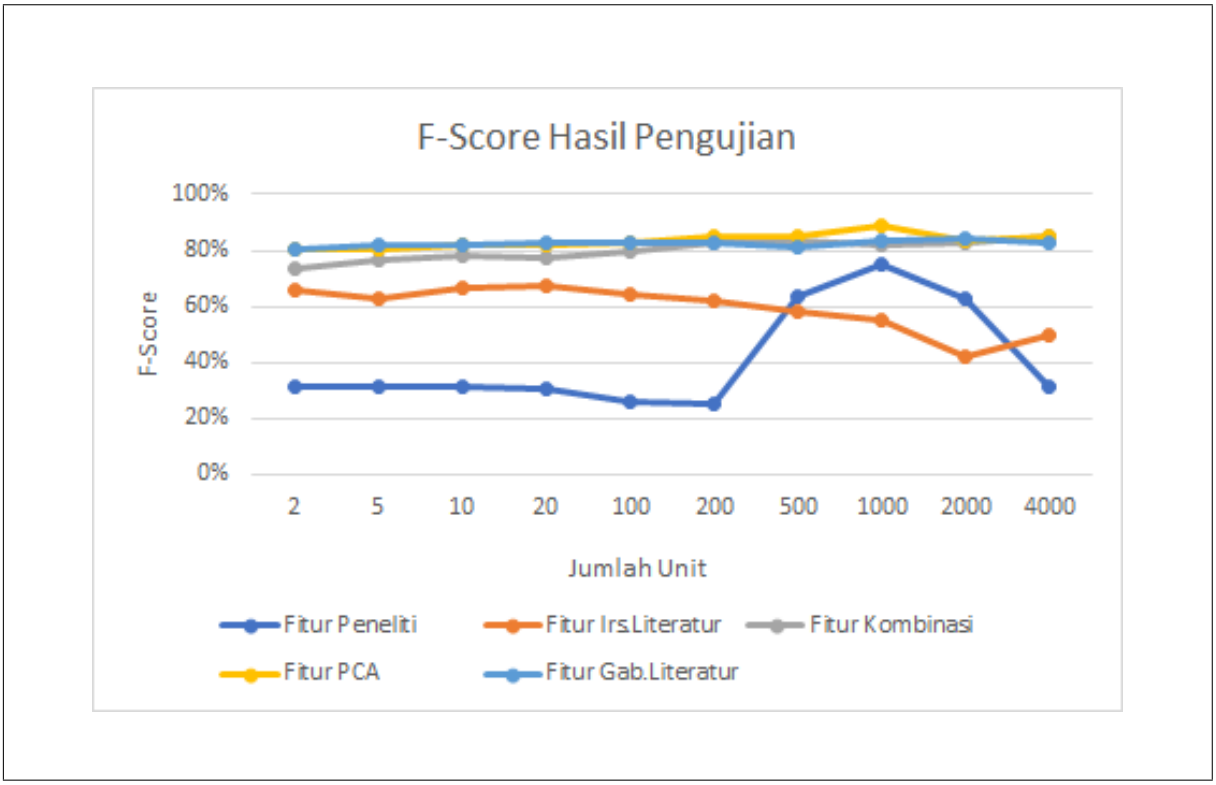
Berikut grafik hasil akurasi pengujian seluruh set data.



Gambar 4.8 *Grafik Akurasi Hasil Pengujian*

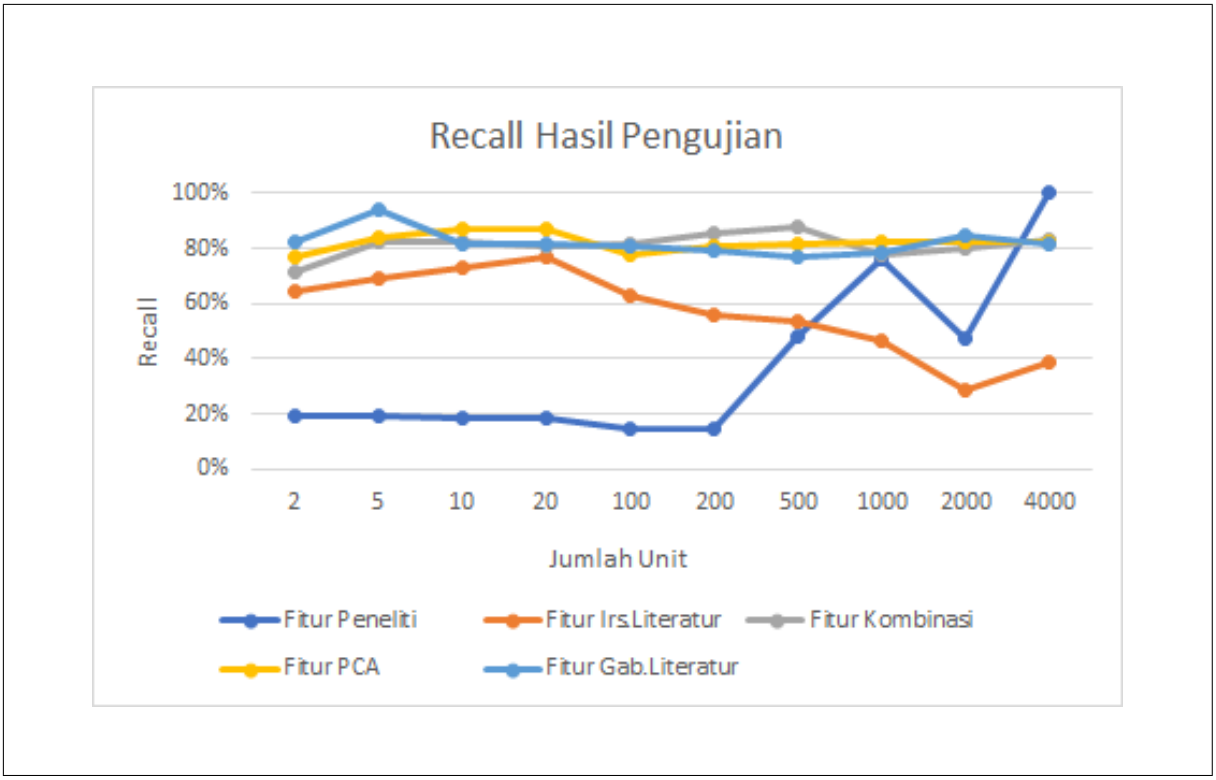
Selain akurasi, penelitian ini akan dianalisis mempertimbangkan nilai *Recall* dan *F-score*. Pada Tabel 4.66, nilai *recall* tertinggi adalah 94.35698% yang diperoleh pada pengujian fitur gabungan literatur dengan jumlah unit 5, 10 *epoch* dan *learning rate* 0.1. Nilai *F-score* tertinggi adalah 88.68530% yang diperoleh pada pengujian terhadap fitur PCA dengan jumlah unit 1000, 500 *epoch*, dan *learning rate* 0.01.

Berikut merupakan grafik hasil pengujian berdasarkan nilai *F-Score*.



Gambar 4.9 *Grafik F-Score Hasil Pengujian*

Berikut merupakan grafik hasil pengujian berdasarkan nilai *Recall*.



Gambar 4.10 *Grafik Recall Hasil Pengujian*

Parameter variasi metode pemilihan fitur dari dataset yang digunakan memberikan

perubahan yang paling signifikan terhadap tingkat akurasi dan *recall* hasil pengujian. Hasil pengujian terhadap fitur peneliti yang berjumlah 8 fitur dapat mencapai akurasi 90.58680% dengan nilai *recall* tertinggi 76.34090%. Fitur irisan literatur yang berjumlah 9 fitur mencapai akurasi 87.66125% dengan nilai *recall* tertinggi 64.82618%. Fitur kombinasi yang berjumlah 17 fitur menghasilkan hasil pengujian dengan akurasi 96.46730% dan *recall* 82.83046%. Hasil pengujian terhadap fitur gabungan literatur sejumlah 27 fitur mencapai akurasi tertinggi 92.33980% dengan *recall* 94.35698%. Metode pemilihan fitur terakhir yaitu dengan PCA sejumlah 30 fitur menghasilkan akurasi 96.10441% dengan *recall* 82.05174%. Berdasarkan hasil tersebut, peneliti menyimpulkan bahwa metode pemilihan fitur berpengaruh terhadap hasil pengujian, namun jumlah fitur yang dipilih tidak selalu berbanding lurus dengan nilai akurasi yang dihasilkan.

Pengaruh pemilihan parameter jumlah unit LSTM memberikan hasil yang fluktuatif. Pengujian yang dilakukan terhadap fitur kombinasi dan PCA, menunjukkan semakin tinggi jumlah unit, semakin baik akurasi yang dapat dihasilkan. Namun, pada pengujian terhadap fitur literatur dan peneliti, semakin tinggi jumlah unit LSTM, menghasilkan penurunan nilai akurasi dan *recall*. Peneliti menyimpulkan bahwa jumlah unit tidak berbanding lurus dengan nilai *recall* dan akurasi hasil pengujian sehingga dibutuhkan beberapa contoh pengujian untuk mengetahui penggunaan jumlah unit terbaik.

Jumlah *epoch* merupakan parameter yang menunjukkan jumlah pengulangan pembelajaran seluruh *dataset*. Berdasarkan analisis data akurasi yang dihasilkan pada tahap pengujian dengan beberapa variasi *epoch*, dapat disimpulkan bahwa pertambahan jumlah *epoch* berbanding lurus dengan peningkatan nilai akurasi. Hal ini terlihat dari hasil pengujian pada fitur kombinasi terhadap seluruh unit LSTM dengan *epoch* 1 memiliki rentang akurasi antara 88.12362% - 91.87485%, sedangkan hasil pengujian dengan *epoch* 10 memiliki rentang akurasi 89.91766% - 94.46730%. Begitupun dengan pengujian terhadap fitur lain yang menghasilkan hasil yang semakin baik.

Pengaruh parameter *learning rate* terhadap hasil pengujian memberikan hasil yang berbeda pada konfigurasi yang berbeda. Hal ini terlihat pada pengujian terhadap fitur PCA dengan 4000 unit LSTM, pengaplikasian nilai *learning rate* 0.01 menghasilkan akurasi yang lebih tinggi dibandingkan *learning rate* 0.1. Sedangkan pengujian pada fitur gabungan dengan 1000 unit LSTM, penggunaaan nilai *learning rate* 0.5 menghasilkan akurasi yang lebih rendah dibandingkan *learning rate* 0.1. Peneliti menyimpulkan, bahwa diperlukan pengujian untuk menentukan nilai *learning rate* terbaik yang sesuai dengan karakteristik set data yang digunakan.

Grafik pada Gambar 4.10 nilai *recall* tertinggi diperoleh oleh pengujian terhadap fitur gabungan literatur dengan nilai 94.35698%. Nilai *recall* yang tinggi menunjukkan tingginya nilai *true positive* dan rendahnya nilai *false negative* dalam mendeteksi *malware botnet*, sehingga merupakan menjadi salah satu fokus hasil pengujian yang penting dalam menentukan

kesimpulan.

Berdasarkan hasil pengujian, nilai akurasi tertinggi diperoleh oleh pengujian terhadap metode pemilihan fitur PCA dengan akurasi 96.10441% dan nilai *recall* 82.06867%. Nilai *recall* tertinggi diperoleh oleh pengujian terhadap fitur gabungan literatur dengan nilai *recall* 94.35698% dan akurasi 92.33980%. Berdasarkan hasil tersebut, mempertimbangkan selisih akurasi sebesar 3.76461 sedangkan selisih *recall* sebesar 12.28831, peneliti menyimpulkan hasil pengujian terbaik diperoleh melalui pengujian menggunakan fitur gabungan literatur.

Dari seluruh hasil pengujian dan analisisnya, peneliti menyimpulkan bahwa metode pemilihan fitur terbaik pada penelitian ini adalah dengan menggunakan metode gabungan literatur dengan jumlah fitur sebanyak 27, yang dilakukan terhadap parameter LSTM sebagai berikut; 5 jumlah unit, 10 *epoch*, dan menggunakan nilai *learning rate* 0.1.

BAB V

PENUTUP

5.1 Kesimpulan

Kesimpulan dari pembuatan sistem pendeteksian malware botnet dengan menggunakan metode *Long Short Term Memory* berdasarkan fitur *network traffic* melalui pengujian yang telah dilakukan adalah sebagai berikut:

1. Penelitian ini berhasil menerapkan *Recurrent Neural Network* dengan *Long Short Term Memory* untuk mendeteksi *malware botnet* terhadap fitur *network traffic* dengan akurasi 92.33980% dan *recall* 94.35698%.

2. Metode pemilihan fitur dari *network traffic* memiliki pengaruh besar dalam menentukan kualitas proses pembelajaran yang divalidasi oleh hasil pengujian. Dalam penelitian ini, fitur terbaik untuk proses pembelajaran mesin, dihasilkan menggunakan fitur gabungan literatur sejumlah 27 fitur yang menggabungkan set fitur dari beberapa penelitian sebelumnya.

3. Berdasarkan hasil pengujian yang dilakukan pada penelitian ini, konfigurasi parameter LSTM memiliki pengaruh yang signifikan terhadap akurasi dan *recall* hasil pengujian. Hasil pengujian terbaik diperoleh dengan konfigurasi 5 unit LSTM, 10 *epoch* dan *learning rate* 0.1 terhadap fitur gabungan literatur.

5.2 Saran

Saran untuk pengembangan yang dilakukan untuk sistem pendeteksian *malware botnet* dengan menggunakan metode *Long Short Term Memory* berdasarkan fitur *network traffic* adalah sebagai berikut:

1. Menggunakan variasi lain dari RNN LSTM seperti menambahkan *peephole connection* dan *Gated Recurrent Unit*.

2. Mengaplikasikan konfigurasi model LSTM dengan dataset yang berbeda dan parameter yang lebih bervariasi untuk mencapai hasil yang lebih akurat.

3. Menggunakan variasi pendekatan lain dalam proses ekstraksi fitur untuk menghasilkan set data dengan fitur yang lebih baik.

DAFTAR PUSTAKA

[1] David Zhao, Issa Traore, Bassam Sayed, and *. ”Botnet Detection Based on Traffic Behaviour Analysis and Flow Intervals”, *Computers and Security 39:2-16*, 2013.

[2] Matija Stevanovic and Jens Myrup Pedersen, ”An Analysis of Network Traffic Classification for Botnet Detection”, IEEE, 2015.

[3] Pablo Torres, Carlos Catania, Sebastian Garcia and Carlos Garcia, 2016. ”An Analysis of Recurrent Neural Networks for Botnet Detection Behaviour”, IEEE, 2016.

[4] Sumith Maniath, Aravind Ashok, and *. 2017. ”Deep Learning LSTM Based Ransomware Detection”,IEEE, 2017.

[5] Sepp Hochreiter, Jurgen Schmidhuber. 1997. ”Long Short Term Memory”, *Neural Computation 9*, no. 8: 1735-1780, 1997.

[6] Tomas Mikolov, and *. 2010. ”Recurrent Neural Network Based Language Model”, *Eleventh annual conference of the international speech communication association*, 2010.

[7] R. Tariq. ”Make Your Own Neural Network”, *Independent Publishing Platform*, pp. 56-74, 2016.

[8] Alex Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*, Illustrated Edition. Springer, 2012

[9] spamhaus.org, ”Spamhaus Botnet Threat Report 2017”, 2018.[Online]. Available: https://www.spamhaus.org/news/article/772/spamhaus-botnet-threat-report-2017

[10] Ms.Amruta Kapre and Mrs. B. Padmavathi, ”Adaptive Behaviour Pattern Based Botnet Detection Using Traffic Analysis and Flow Intervals”, IEEE, 2017.

[11] Sherif Saad, Issa Traore, and *., ”Detecting P2P Botnets through Network Behavior Analysis and Machine Learning”, IEEE, 2014.

[12] Matija S and Jens M, ”An Efficient Flow-Based Botnet Detection Using Supervised Maching Learning”, IEEE, 2014

[13] B. Denny. (2015, October 8). Recurrent Neural Networks Tutorial, Part 3 – Backpropagation Through Time and Vanishing Gradients [Online]. Available: http://www.wildml.com/2015/10/recurrent-neural-networks-tutorial-part-3-backpropagation-through-time-and-vanishing-gradients/

[14] Bontemps, Loic, James McDermott, and Nhien-An Le-Khac. ”Collective anomaly detection based on long short-term memory recurrent neural networks”, *In International Conference on Future Data and Security Engineering*, pp. 141-152. Springer, Cham, 2016.

[15] E. Alpaydin. ”Introduction to Machine Learning – Second Edition,” The MIT Press, 2010.

[16] R. Sunil. (2017, May 29). Understanding and coding Neural Networks From Scratch in Python and R [Online]. Available: https://www.analyticsvidhya.com/blog/2017/05/neuralnetwork- from-scratch-in-python-and-r/

[17] G. Klaus, Rupesh K. Srivastava, K. Jan, Bas R. Steunebrink, S. Jurgen. (2017, October 4). LSTM: A Search Space Odyssey. TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. arXiv:1503.04069v2.

[18] Ian G., Yoshua B., Aaron C., *Deep Learning*. MIT Press, 2016.

[19] Jolliffe.I.T., *Principal Component Analysis*. 2nd ed. Springer-Verlag New York, 2002.

[20] Brase CH, *Understanding Basic Statistics, Enchanced*. Cencage Learning, 2016.

[21] Cyril Goutte and Eric Gaussier, *A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation*. Springer, Berlin, Heidelberg, 2005.

LAMPIRAN

LAMPIRAN A

FITUR NETWORK TRAFFIC

Tabel A-1 Daftar 84 fitur pada dataset

No	Nama Fitur	Deskripsi Fitur
1	Flow ID	ID flow
2	Source IP	Alamat IP flow
3	SrcPort	Port Pengirim
4	Destination IP	Alamat IP tujuan
5	Destination Port	Port tujuan
6	Protocol	Nomor protokol layer transport
7	Timestamp	Waktu paket direkam
8	Flow Duration	Total durasi flow
9	Total FWD Packets	Jumlah paket yang dikirim
10	Total BWD Packets	Jumlah paket yang diterima
11	Total Length of FWD Packets	Total panjang paket yang dikirim
12	Total Length of BWD Packets	Total panjang paket yang diterima
13	FWD Packet Length Max	Nilai maksimal panjang paket yang dikirim
14	FWD Packet Length Min	Nilai minimal panjang paket yang dikirim
15	FWD Packet Length Mean	Rata-rata panjang paket yang dikirim
16	FWD Packet Length Std	Standar deviasi panjang paket yang dikirim
17	BWD Packet Length Max	Nilai maksimal panjang paket yang diterima
18	BWD Packet Length Min	Nilai minimal panjang paket yang diterima
19	BWD Packet Length Mean	Rata-rata panjang paket yang diterima
20	BWD Packet Length Std	Standar deviasi ukuran panjang yang diterima
21	Flow Bytes/s	Jumlah byte per detik dalam satu flow

22	Flow Packets/s	Jumlah paket per detik dalam satu flow
23	Flow IAT Mean	Nilai rata-rata jarak waktu antar paket
24	Flow IAT Std	Standar deviasi jarak waktu antar paket
25	Flow IAT Max	Nilai maksimal jarak waktu antar paket
26	Flow IAT Min	Nilai minimal jarak waktu antar paket
27	FWD IAT Total	Total jarak waktu pengiriman antar paket
28	FWD IAT Mean	Rata-rata jarak waktu pengiriman antar paket
29	FWD IAT Std	Standar deviasi jarak waktu pengiriman antar paket
30	FWD IAT Max	Nilai maksimal jarak waktu pengiriman antar paket
31	FWD IAT Min	Nilai minimal jarak waktu pengiriman antar paket
32	BWD IAT Total	Total jarak waktu penerimaan antar paket
33	BWD IAT Mean	Rata-rata jarak waktu penerimaan antar paket
34	BWD IAT Std	Standar deviasi jarak waktu penerimaan antar paket
35	BWD IAT Max	Nilai maksimal jarak waktu penerimaan antar paket
36	BWD IAT Min	Nilai minimal jarak waktu penerimaan antar paket
37	FWD PSH Flags	Jumlah Pushing Flag saat pengiriman paket
38	BWD PSH Flags	Jumlah Pushing Flag saat penerimaan paket
39	FWD URG Flags	Jumlah Urgent Flag saat pengiriman paket
40	BWD URG Flags	Jumlah Urgent Flag saat penerimaan paket
41	FWD Header Length	Panjang header paket yang dikirim
42	BWD Header Length	Panjang header paket yang diterima
43	FWD Packets/s	Jumlah paket yang dikirim setiap detik
44	BWD Packets/s	Jumlah paket yang diterima setiap detik
45	Min Packet Length	Panjang minimal paket dalam flow
46	Max Packet Length	Panjang maksimal paket dalam flow
47	Packet Length Mean	Rata-rata panjang paket dalam flow

48	Packet Length Std	Standar deviasi panjang paket dalam flow
49	Packet Length Variance	Nilai varians panjang paket dalam flow
50	FIN Flag Count	Jumlah Finish Flag
51	SYN Flag Count	Jumlah Synchronize Flag
52	RST Flag Count	Jumlah Reset Flag
53	PSH Flag Count	Jumlah Push Flag
54	ACK Flag Count	Jumlah Acknowledged Flag
55	URG Flag Count	Jumlah Urgent Flag
56	CWE Flag Count	Jumlah CWR TCP Flag
57	ECE Flag Count	Jumlah ECE TCP Flag
58	Down/Up Ratio	Rasio upload dan download
59	Average Packet Size	Rata-rata ukuran paket
60	Avg FWD Segment Size	Rata-rata ukuran segmen yang dikirim
61	Avg BWD Segment Size	Rata-rata ukuran segmen saat diterima
62	FWD Avg Bytes/Bulk	Rata-rata ukuran bulk bytes yang dikirim
63	FWD Avg Packets/Bulk	Rata-rata ukuran bulk paket yang dikirim
64	FWD Avg Bulk Rate	Rata-rata jumlah bulk yang dikirim
65	BWD Avg Bytes/Bulk	Rata-rata ukuran bulk bytes yang diterima
66	BWD Avg Packets/Bulk	Rata-rata ukurang bulk paket yang diterima
67	BWD Avg Bulk Rate	Rata-rata jumlah bulk yang diterima
68	Subflow FWD Packets	Rata-rata jumlah paket dalam setiap subflow yang dikirim
69	Subflow FWD Bytes	Rata-rata jumlah bytes dalam setiap subflow yang dikirim
70	Subflow BWD Packets	Rata-rata jumlah paket dalam setiap subflow yang diterima
71	Subflow BWD Bytes	Rata-rata jumlah bytes dalam setiap subflow yang diterima
72	Init Win Bytes FWD	Jumlah bytes yang dikirim pada window awal
73	Init Win Bytes BWD	Jumlah bytes yang diterima pada window awal

74	Act Data Packet FWD	Jumlah paket yang dikirim setiap payload data TCP
75	Min Seg Size FWD	Nilai minimal ukuran segment yang dikirim
76	Active Mean	Rata-rata waktu flow aktif
77	Active Std	Standar deviasi waktu flow aktif
78	Active Max	Nilai maksimal waktu flow aktif
79	Active Min	Nilai minimal waktu flow aktif
80	Idle Mean	Rata-rata waktu flow idle
81	Idle Std	Standar deviasi waktu flow idle
82	Idle Max	Nilai maksimal waktu flow idle
83	Idle Min	Nilai minimal waktu flow idle
84	Label	Status flow (botnet atau benign)

LAMPIRAN B

FITUR PENELITIAN TERKAIT

Tabel B-1 Daftar Fitur Jurnal Penelitian [1]

No	Feature Name	Description
1	SrcIP	Flow source IP address
2	SrcPort	Flow source Port address
3	DstIP	Flow destination IP address
4	DstPort	Flow destination Port address
5	Protocol	Transport layer protocol
6	Duration	Flow duration
7	APL	Average payload packet length for time interval
8	FWD Pack	Total of payload packet on forward direction
9	Bwd Pack	Total of payload packet on backward direction
10	PV	Variance of payload packet length for time interval
11	PX	Number of packets exchanged
12	PPS	Number of packets exchanged per second
13	BPS	Number of bytes exchanged per second
14	FPS	Size of the first packet in flow
15	TBP	The average time between packets
16	IAT Std	Standard deviation of payload packet length for time interval
17	NR	Number of reconnects for a flow
18	FPH	Number of flows from this address over total flows

Tabel B-2 Daftar Fitur Jurnal Penelitian [11]

No	Feature Name	Description
1	SrcPort	Flow source Port address
2	Pack Length	Payload size in bytes
3	APL	Average packet length per flow
4	TPC	Total number of packets per flow
5	TBT	Total number of bytes per flow
6	IOP	The ratio between number of incoming over outgoing packets
7	DPL	The total number of subsets of packets of the same length over the total number of packets in the same flow
8	PL	The total number of bytes of all the packets over the total number of packets in the same flow
9	SPDP	Ratio between the number of source ports to the number of destination ports
10	CDA	Number of connections over the number of destination address
11	TPDA	The sum of different transmission protocol
12	DASP	The number of destination IP connected to the same open port

Tabel B-3 Daftar Fitur Jurnal Penelitian [12]

No	Feature Name	Description
1	SrcPort	Flow source Port address
2	L3/L4 Protocol Identifier	Total number of protocol L3/L4
3	L7 Protocol Identifier	Total number of protocol L7
4	Total Payload Length in Byte	Total payload length in byte
5	Packet Byte Mean	Average of byte in one packet
6	Packet Byte Median	Median of byte in one packet
7	Packet Byte Std	Standard deviation of byte in one packet
8	Percentage of Packet	Percentage of packet size
9	TCP Flags	Number of TCP flags
10	ICMP Flags	Number of ICMP flags
11	Flow Duration	Flow duration
12	IAT Mean	Average of time between packets
13	IAT Median	Median of time between packets
14	IAT Std	Standard deviation of time between packets
15	Ratio Packet IN/OUT	Number of packet in/out ratio
16	Ratio Byte IN/OUT	Number of byte in/out ratio
17	Ratio IAT IN/OUT	Number of time between packet ratio

LAMPIRAN C

SKENARIO PENGUJIAN

Tabel C-1 Skenario Pengujian

No.	Unit LSTM	Epoch	Learning Rate
1	2	1	0.1
2	2	2	0.1
3	2	3	0.1
4	2	10	0.1
5	2	1	0.5
6	2	2	0.5
7	2	3	0.5
8	2	10	0.5
9	5	1	0.1
10	5	2	0.1
11	5	3	0.1
12	5	10	0.1
13	5	1	0.5
14	5	2	0.5
15	5	3	0.5
16	5	10	0.5
17	10	1	0.1
18	10	2	0.1
19	10	3	0.1

20	10	10	0.1
21	10	1	0.5
22	10	2	0.5
23	10	3	0.5
24	10	10	0.5
25	20	1	0.1
26	20	2	0.1
27	20	3	0.1
28	20	10	0.1
29	20	1	0.5
30	20	2	0.5
31	20	3	0.5
32	20	10	0.5
33	100	1	0.1
34	100	2	0.1
35	100	3	0.1
36	100	10	0.1
37	100	1	0.5
38	100	2	0.5
39	100	3	0.5
40	100	10	0.5
41	200	1	0.1
42	200	2	0.1
43	200	3	0.1

44	200	10	0.1
45	200	1	0.5
46	200	2	0.5
47	200	3	0.5
48	200	10	0.5
49	500	1	0.1
50	500	2	0.1
51	500	3	0.1
52	500	10	0.1
53	500	1	0.5
54	500	2	0.5
55	500	3	0.5
56	500	10	0.5
57	1000	1	0.1
58	1000	2	0.1
59	1000	3	0.1
60	1000	10	0.1
61	1000	1	0.5
62	1000	2	0.5
63	1000	3	0.5
64	1000	10	0.5
65	2000	1	0.1
66	2000	2	0.1
67	2000	3	0.1

68	2000	10	0.1
69	2000	1	0.5
70	2000	2	0.5
71	2000	3	0.5
72	2000	10	0.5
73	4000	1	0.1
74	4000	2	0.1
75	4000	3	0.1
76	4000	10	0.1
77	4000	1	0.5
78	4000	2	0.5
79	4000	3	0.5
80	4000	10	0.5