Lembar Kemajuan Tesis

IDENTITAS

Nama	Tomi Yahya C
NIM	176150100111006
No HP	08563529388
Email	tomi@rekavisitama.net
Pembimbing 1	Wayan Firdaus Mahmudy, S.Si.,M.T., Ph.D.
Pembimbing 2	Dr.Eng. Ahmad Afif Supianto, S.Si, M.Kom.
Judul	Sistem Deteksi Intrusi Jaringan Dengan Metode Restricted Growing Self-Organizing Map

KEMAJUAN

Tanggal Mulai	28 Agustus 2018
Tanggal Ujian Proposal	27 September 2018
Tanggal Seminar Hasil	28 Desember 2018
Tanggal Ujian Akhir	18 April 2019

PUBLIKASI SELAMA STUDI S2

No	Penulis	Jurnal/ Conference	Judul Artikel	Status
1	Tomi Yahya C. Wayan F Mahmudy	Sustainable Information Engineering and Technology (SIET 2018)	Text Classification and Visualization on News Title Using Self Organizing Map	Published
2	Tomi Yahya C. M Syauqi Haris Rafiuddin Rody Wayan F Mahmudy	Sustainable Information Engineering and Technology (SIET 2018)	Optimization of Fuzzy Time Series Interval Length Using Modified Genetic Algorithm	Published
3	Tomi Yahya C. Ahmad Afif Supianto Wayan F Mahmudy	Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)	Anomaly-Based Intrusion Detector System Using Restricted Growing Self Organizing Map	Published

BIMBINGAN

Tanggal	Pembimbing	Perbaikan
28 Agustus 2018	1 (tatap Muka)	Judul dan Latar belakang (contoh kasus)
10 September 2018	2 (tatap Muka)	Rumusan masalah
13 September 2018	1 (email)	Penjelasan tentang RGSOM + revisi minor penulisan

19 September 2018	1 (tatap muka)	Pada 2.5 GSOM disebutkan kelemahannya dan dijelaskan usaha untuk mengatasinya akan dijelaskan di metode RGSOM yang akan dijelaskan di bab 3.
25 November 2018	2 (tatap muka)	Peningkatan akurasi metode RGSOM, Pengujian metode GSOM
19 Desember 2018	2 (Tatap Muka)	Penambahan tentang PCA
20 Desember 2018	1 (Email)	Perbaikan minor dan pendaftaran seminar hasil
28 Desember 2018	2 (Tatap Muka)	Latarbelakang diperjelas, Pengujian parameter ditampilkan waktunya. PCA dan SVM disinggung dalam latar belakang kenapa digunakan sebagai pembanding.
28 Desember 2018	1 (Tatap muka)	Pengujian parameter menggunakan K-Fold
24 Januari 2019	1 (Tatap muka)	Gambar pengukuran akurasi dan waktu komputasi di sajikan secara terpisah.
31 Januari 2019	1 (Tatap muka)	Hasil pengujian parameter dengan K-Fold.
14 Februari 2019	1 (Tatap muka)	Melanjutkan untuk proses final tesis
15 Februari 2019	2 (Tatap muka)	Redaksional Rumusan masalah, PCA belum dimunculkan pada BAB 1,
18 Februari 2019	2 (Tatap muka)	Restruktur rumusan masalah, menyinggung penggunaan PCA pada latar belakang, menambah pengujian untuk RGSOM tanpa reduksi fitur untuk mengetahui tujuan penggunaan PCA apakah berpengaruh terhadap waktu komputasi. Sub bab pada pembahasan mengacu pada rumusan masalah.
12 Maret 2019	1 (Email)	Hasil pengujian RGSOM tanpa Reduksi Fitur
15 Maret 2019	2 (Tatap muka)	Finalisasi tesis
5 April 2019	1 (Tatap muka)	Lampiran, dan finalisasi laporan tesis

SISTEM DETEKSI INTRUSI JARINGAN DENGAN METODE *RESTRICTED GROWING SELF-ORGANIZING MAP*

TESIS

Untuk memenuhi sebagian persyaratan
memperoleh gelar Magister Komputer

Disusun oleh:
Tomi Yahya Christyawan
NIM: 176150100111006

PROGRAM STUDI MAGISTER ILMU KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

SISTEM DETEKSI INTRUSI JARINGAN DENGAN METODE RESTRICTED GROWING SELF-ORGANIZING MAP

TESIS

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Magister Komputer

Disusun oleh:
Tomi Yahya Christyawan
NIM: 176150100111006

Tesis ini telah diuji dan dinyatakan lulus pada
18 April 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D.	Dr.Eng. Ahmad Afif Supianto, S.Si, M.Kom
NIP. 19720919 199702 1 001	NIP. 201201 820623 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T, Ph.D.
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah tesis ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah tesis ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia tesis ini digugurkan dan gelar akademik yang telah saya peroleh (Magister) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang,

Tomi Yahya Christyawan
NIM: 176150100111006

KATA PENGANTAR

Segala puji syukur kami berikan kepada Tuhan Yang Maha Esa atas rahmat dan berkatnyaNya sehinga penulis dapat menyelesaikan Tesis yang berjudul "**SISTEM DETEKSI INTRUSI JARINGAN DENGAN METODE RESTRICTED GROWING SELF-ORGANIZING MAP**". Tesis ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Magister Komputer di Fakultas Ilmu Komputer Universitas Brawijaya Malang.

Melalui kesempatan ini, penulis ingin menyampaikan rasa hormat dan terima kasih penulis yang sebesar - besarnya kepada semua pihak yang telah memberikan bantuan dalam penulisan tesis ini. Penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D. selaku dosen pembimbing I, yang telah memberikan waktu, ilmu dan saran untuk menyelesaikan tesis ini.
2. Bapak Dr.Eng. Ahmad Afif Supianto, S.Si, M.Kom selaku dosen pembimbing II, yang telah memberikan waktu, ilmu dan saran untuk menyelesaikan tesis ini.
3. Bapak Dr. Eng. Fitri A. Bachtiar, S.T, M.Eng. selaku dosen penguji I dan Ketua Program Studi Magister Ilmu Komputer yang telah memberikan masukan dalam penyelesaian tesis ini.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku dosen penguji II dan Ketua Jurusan Teknik Informatika yang telah memberikan saran dalam penulisan tesis ini.
5. Keluarga besar Magister Ilmu Komputer yang telah memberikan dukungan dan bantuan selama masa kuliah.
6. Keluarga yang mendoakan dan telah memberikan dukungan dan semangat kepada penulis untuk menyelesaikan laporan tesis ini.

Penulis mengucapkan terima kasih dan permohonan maaf apabila dalam penyusunan laporan ini terdapat banyak kekurangan. Penulis berharap ada saran maupun kritik yang berguna untuk pengembangan di masa yang akan datang. Semoga tesis ini dapat memberikan manfaat bagi semua pihak.

Malang, Mei 2019

Penulis

tomi@rekavisitama.net

ABSTRAK

Perkembangan teknologi internet dan jaringan yang pesat juga diikuti ancaman berbahaya dari serangan terhadap jaringan dan komputer. Sistem Deteksi Intrusi atau *Intrusion Detection System* (IDS) dibuat untuk menangani permasalahan tersebut. Pengembangan IDS saat ini banyak menggunakan *machine learning* untuk melakukan klasifikasi terhadap serangan berbahaya. Salah satu metode untuk melakukan klasifikasi dan visualisasi adalah dengan *Self-Organising Map* (SOM). SOM mampu melakukan klasifikasi dan visualisasi pada proses pembelajaran, sehingga dapat digunakan untuk memperoleh pengetahuan baru. Permasalahan yang dimiliki oleh SOM yaitu waktu komputasi yang kurang efisien apabila diterapkan pada data yang berukuran besar (*big data*), permasalahan ini telah di coba diselesaikan dengaan metode Growing Self-Organizing Map (GSOM) namum masih belum optimal. Pada penelitian ini diajukan metode *Restricted Growing SOM* (RGSOM) yang dikombinasikan dengan metode *Principle Component Analysis* (PCA) sebagai reduksi fitur yang bertujuan untuk meningkatkan efisiensi waktu komputasi. Proses utama dalam RGSOM adalah proses berkembangnya map dan proses pembaharuan bobot pada map. Berkembangnya map pada RGSOM dibatasi oleh jumlah node maksimum dan ambang batas berkembang (*Growing Threshold*). Pada penelitian sebelumnya metode *Support Vector Machine* (SVM) dan hibridisasinya menghasilkan akurasi yang tinggi, sehingga dalam penelitian ini SVM akan digunakan sebagai metode pembanding sebagai referensi akurasi dari metode yang diusulkan. Metode yang diusulkan dalam penelitian ini mampu menghasilkan nilai akurasi sebesar 99.53 % dan waktu komputasi selama pelatihan adalah 519.4 detik dan pengujian adalah 884.8 detik, dan paling efisien dari pada metode pembanding lainnya (SOM, GSOM, dan SVM). Pada penelitian ini metode yang diusulkan mampu meningkatkan efisiensi waktu komputasi dalam melakukan klasifikasi tanpa mengorbankan akurasi.

Kata kunci: *self-organizing map, restricted growing self-organizing map, neural network, principle component analysis, klasifikasi, visualisasi, reduksi fitur, big data.*

ABSTRACT

The growth of internet technology and networks also involves threats to networks and computers. Intrusion Detection System (IDS) is made to solve this problem. IDS development currently uses many learning machines to classify dangerous attacks. One method for classification and visualization is the Self-Organizing Map (SOM). SOM is able to classify and visualize the learning process, so that it can be used to gain new knowledge. The problem that is owned by SOM is that computing time is less efficient when applied to big data, this problem has been tried to be solved with the Growing Self-Organizing Map (GSOM) method, but it is still not optimal. In this study is proposed the Restricted Growing SOM (RGSOM) method combine with the Principle Component Analysis (PCA) method as a feature reduction that aims to improve computational time efficiency. The main process in RGSOM is the process of growing maps and weight renewal processes. The map growth on RGSOM limited by the maximum number of nodes and growing threshold. In the previous study for IDS method, Support Vector Machine (SVM) and hybridization methods produced high accuracy to classify attacks, so SVM will be used as a comparison method as a reference of accuracy from the proposed method. The method proposed in this study reached 99.53% and the calculation time for training was 519.4 seconds and the test were 884.8 seconds, and the most efficient of the other comparison methods (SOM, GSOM, and SVM). In this study the proposed method is able to improve computational time efficiency in conducting classifications without sacrificing accuracy.

Keywords: *self-organizing map, restricted growing self-organizing map, neural network, principle component analysis, classification, visualization, feature reduction, big data*.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Ancaman keamanan terhadap penggunaan internet dan jaringan komputer semakin lama semakin meningkat. Almeida, Doneda, dan de Souza Abreu (2017) dalam sebuah papernya membahas tentang konsep dan cara praktis untuk melakukan peperangan terhadap kejahatan siber dan dampaknya pada dunia maya. Almeida mencontohkan apa yang terjadi pada Dyn, sebuah perusahaan yang melakukan kontrol terhadap infrastruktur *domain name system* (DNS), yang pada tanggal 21 Oktober 2016 mengalami serangan DDoS (*Distributed Denial-of-Service*) oleh *botnet* yang dinamakan Mirai. Serangan DDoS yang dialami oleh Dyn membuat beberapa situs antara lain Twitter, the Guardian, Netflix, Reddit, CNN dan banyak lagi lainnya yang berada di Eropa dan US tidak dapat diakses. Berdasarkan penyelidikan, *botnet* Mirai menginfeksi perangkat "*internet of things*" (IoT) seperti kamera digital dan *Digital Video Recorder* (DVR). Dengan semakin banyaknya perangkat yang terkoneksi melalui internet dan ketergantungan ekonomi, militer, dan pemerintahan negara pada dunia maya, hal ini membuat internet semakin rentan terhadap serangan siber.

Serangan siber terbesar pada tahun 2018 dengan kategori DDoS terjadi pada tanggal 28 Februari 2018 yang menargetkan situs Github (Skottler, 2018). Serangan siber tersebut mampu membuat situs Github tidak dapat diakses selama kurang dari 10 menit. Github mencatat bahwa ada *inbound traffic* sebesar 1,35 Tbps pada saat terjadinya serangan. Serangan ini tentu saja merugikan banyak pihak karena Github merupakan salah satu situs terbesar di dunia yang menyediakan layanan manajemen source code, yang memiliki beberapa klien perusahan besar dunia.

Live Cyber Attack Threat Map (Checkpoint, 2018), sebuah situs yang dikelola oleh checkpoint dengan bekerja sama dengan ThreatCloud Intelligence, menampilkan data pada tanggal 1 september 2018 terdapat 11,047,067 serangan terhadap jaringan komputer di seluruh dunia. Pada hari tersebut Indonesia menduduki peringkat ke tiga sebagai negara yang menjadi target serangan siber tertinggi di dunia. Urutan negara-negara yang mengalami serangan siber tersebut adalah India, USA, Indonesia, Australia, Mexico, United Kingdom, Brasil, Norway, Philippines, dan China.

Beberapa jenis serangan baru terhadap jaringan muncul secara berkala, sehingga memunculkan suatu tantangan untuk mengembangkan metode pengamanan jaringan secara fleksibel dan adaptif. Metode pengamanan jaringan yang fleksibel dan adaptif ini diperlukan karena deteksi terhadap koneksi pada jaringan yang mencurigakan dan penggunaan komputer tidak dapat dilakukan oleh *firewall* konvensional. *Firewall* merupakan sitem yang berfungsi untuk membatasi akses yang tidak diharapkan dari atau ke dalam suatu jaringan internal. Alat yang berupa perangkat keras atau perangkat lunak untuk mendeteksi intrusi yang berbahaya disebut Sistem Deteksi Intrusi atau IDS.

IDS dibagi menjadi dua bagian *Network IDS* (NIDS) dan *Host-Based IDS* (HIDS). NIDS melakukan analisis paket data yang terdapat pada jaringan dan mengamati kondisi (*signatures*) dari *traffic* jaringan tersebut dan dibandingkan dengan *database* (Vaarandi & Podins, 2010). HIDS diimplementasikan secara langsung pada komputer dan mengamati sistem operasi melalui *log, file system,* dan *hard drives* untuk melakukan deteksi terhadap intrusi. Terdapat dua cara analisis NIDS antara lain melalui *Pattern Matching Based* dan *Anomaly Detection Based*. *Pattern Matching Based* menentukan suatu intrusi dengan cara membandingkan aktivitas dengan *signatures* yang terdapat pada *database*. *Anomaly Detection Based* menentukan intrusi berdasarkan anomali yang terjadi pada *traffic* jaringan (Amato, Mazzocca, Moscato, & Vivenzio, 2017). Teknik deteksi dengan basis anomali ini mampu melakukan deteksi secara fleksibel dan adaptif karena tidak berdasarkan signature tertentu yang sudah terdaftar di database, melainkan berdasarkan pencarian heuristik atau aturan tertentu.

Beberapa pengembangan IDS berbasis deteksi anomali seringkali menggunakan teknik *machine learning*. Teknik *machine learning* memiliki beberapa pendekatan atau metode. Metode yang digunakan untuk deteksi intrusi secara umum dibedakan menjadi klasifikasi yang termasuk dalam rumpun *supervise learning* dan klasterisasi yang masuk dalam kategori *unsupervised learning*. Tetapi ada juga yang melakukan penggabungan antara klasterisasi dan klasifikasi dalam melakukan deteksi terhadap intrusi jaringan.

Pada proses klasifikasi beberapa penelitian terhadap IDS menggunakan *single classifier* seperti *K-Nearest Neighbors* (KNN) yang dilakukan oleh Almalawi et al. (2014), *Naïve Bayes* (NB) (Han et al., 2015), *Decision Tree* (DT) (Agrawal, 2015). Kabir et al. (2018) menggunakan *Least Square Support Vector Machine* (LS-SVM) sebagai metode klasifikasi. Pada penelitian tersebut hanya digunakan sebagian dari data latih (25.843) dan data uji (19.142) dari *Knowledge Discovery and Data Mining* (KDD) 1999. Hasil dari klasifikasi SVM multikelas dengan metode LS-SVM menghasilkan akurasi terbaik sebesar 99,78 %.

Beberapa penelitian menggunakan *Artificial Neural Network* antara lain *Multilayer Perceptron* (Amato et al., 2017), *Backpropagation Neural Network* (Chiba et al., 2018) dan (Al Huda, Mahmudy, & Tolle, 2016). M. Ibrahim. (2013) menggunakan SOM dengan jumlah data latih sebanyak 494.021 dan data uji sejumlah 311.029 menghasilkan tingkat *detection rate* sebesar 92.37%. *Hierarchical Self-Organizing Map* (HSOM) (De la Hoz et al., 2014) menghasilkan akurasi sebesar 99.12% dengan data latih dan data uji menggunakan data dari KDD-NSL.

Penelitian terkait IDS juga dilakukan dengan menggunakan klasterisasi dan *outliner detection* antara lain A-SPOT (Jabez & Muthukumar, 2015), CANN (Lin, Ke, & Tsai, 2015). Proses deteksi intrusi juga dilakukan secara *hybrid* antara lain *Particle Swarm Optimization* (PSO) dengan SVM (Hosseini Bamakan et al., 2016), PSO dengan K-Mean (Karami & Guerrero-Zapata, 2015), DT dengan SVM (Kim, Lee, & Kim, 2014), Algoritma Genetika dan Fuzzy (Hamamoto et al., 2018) , *Multi level SVM dan Extreme Learning Machine* dengan K-Mean (Al-Yaseen, Othman, & Nazri,

2017) menggunakan data dari KDD 1999 dengan jumlah data latih sebanyak 494.021 dan data uji sejumlah 311.029 dan menghasilkan akurasi sebesar 95,75%, *detection rate* sebesar 95,17 % dan *false alarm rate* sebesar 1,87%.

Meskipun perkembangan dalam penelitian IDS dengan pendekatan klasifikasi meningkat, tetapi studi yang melibatkan visualisasi data berukuran besar sangat minim (Ruan et al., 2017). Zichan Ruan dalam penelitiannya tersebut menggunakan data dari KDD 1999 sebagai representasi dari permasalahan visualisasi data berukuran besar. Metode yang digunakan adalah Multi-Dimensional Scaling (MDS) dan Principal Component Analysis (PCA) untuk visualisasi data berukuran besar.

Salah satu metode klasifikasi dan juga data reduksi yang dapat memvisualisasikan proses pembelajaran adalah SOM (Kohonen, 2013). Dalam beberapa penelitian, SOM mampu menyelesaikan permasalahan klasifikasi dengan baik. Namun SOM reguler mengalami kendala dalam efisiensi waktu komputasi pada saat melakukan proses pembelajaran dengan data latih yang besar. Permasalahan tersebut dikarenakan karakteristik dari SOM yang melakukan perhitungan jarak terkecil dalam penentuan *neuron* pemenang (*winner*) dari setiap vektor referensi pada *hidden layer*. Jumlah vector referensi yang membentuk peta topografi pada SOM reguler inilah yang di bandingkan satu persatu mulai dari input data latih pertama, sehingga waktu komputasi pada saat awal sudah lama.

Alahakoon, Halgamuge, dan Srinivasan (2000) mencoba menyelesaikan permasalahan efisiensi tersebut dengan metode *Growing Self Organizing Map* (GSOM). Dalam penelitian sebelumnya, GSOM dapat digunakan untuk membentuk vektor referensi secara bertahap berdasarkan data latih yang akan di proses. Tetapi hal ini hanya mampu mengatasi permasalahan tersebut pada epoch pertama. Pada epoch berikutnya, vektor referensi sudah semakin banyak dan permasalahan efisiensi waktu komputasi akan muncul kembali.

Pada penelitian ini diajukan metode *Restricted Growing Self Organizing Map* (RGSOM) sebagai metode klasifikasi yang digunakan untuk melakukan deteksi intrusi pada jaringan yang dikombinasikan dengan *Principal Component Analysis* (PCA). PCA pada penelitian ini digunakan sebagai metode untuk melakukan reduksi fitur yang bermanfaat untuk meningkatkan efisiensi waktu komputasi (Ikram, 2016). RGSOM (Christyawan, Supianto, & Mahmudy, 2019) dikembangkan berdasarkan GSOM untuk mengatasi permasalahan efisiensi waktu komputasi pada proses pembelajaran dengan data latih yang berukuran besar. Pada RGSOM jumlah vektor referensi yang bertumbuh dibatasi jumlahnya, sehingga ukuran vektor referensi tetap terkontrol walaupun *epoch* meningkat. Karena jumlah vektor referensi terbatas maka sebuah node harus dapat dipilih lebih dari satu kali sebagai *winner* yang disebut *clustering reference vector* (CRV). Parameter yang terdapat pada RGSOM antara lain *learning rate* awal, *learning rate* akhir, *growing threshold, neighboorhod radius*, dan maksimum *epoch* perlu dicari nilai yang tepat agar menghasilkan akurasi dan waktu komputasi yang optimal. Metode RGSOM dengan *clustering reference vector* akan dijelaskan lebih lanjut pada Bab 3.

Berdasarkan penelitian sebelumnya metode SVM dan beberapa hibridisasinya menghasilkan nilai akurasi yang cukup baik antara lain (Al-Yaseen et al., 2017), (Kabir et al., 2018) dengan akurasi sebesar 99,78 %, sehingga pada penelitian ini akan dilakukan perbandingan juga terhadap metode SVM sebagai salah satu pembanding untuk penentuan apakah metode yang diusulkan memiliki akurasi yang cukup baik. RGSOM diharapkan mampu melakukan klasifikasi pada deteksi intrusi jaringan dengan waktu komputasi yang efisien tanpa mengorbankan akurasi.

1.2 Rumusan masalah

Pada penelitian ini rumusan permasalahan adalah sebagai berikut:

1. Berapa nilai parameter yang sesuai dari metode RGSOM agar dapat mencapai hasil yang optimal dalam proses deteksi terhadap intrusi jaringan?
2. Bagaimana perbandingan waktu komputasi pada RGSOM dibandingkan dengan SOM, GSOM, dan SVM?
3. Bagaimana perbandingan akurasi yang dihasilkan RGSOM dibandingkan dengan metode SOM, GSOM, dan SVM?
4. Apa pengaruh reduksi fitur dengan menggunkan PCA terhadap waktu komputasi dan akurasi pada metode RGSOM?
5. Bagaimana visualisasi peta topografi pada metode RGSOM dapat memberikan pengetahuan baru pada proses pembelajaran?
6. Bagaimana kemampuan metode RGSOM dalam melakukan deteksi intrusi jaringan dengan waktu komputasi yang efisien tanpa mengorbankan akurasi melalui proses klasifikasi?

1.3 Tujuan

Penulisan tesis ini bertujuan untuk:

1. Mengetahui nilai parameter yang sesuai agar diperoleh hasil yang optimal dalam melakukan klasifikasi pada deteksi intrusi jaringan.
2. Mengetahui perbandingan waktu komputasi metode RGSOM dengan metode SOM, GSOM dan SVM.
3. Mengetahui perbandingan akurasi yang dihasilkan RGSOM dengan metode SOM, GSOM, dan SVM.
4. Mengetahui pengaruh reduksi fitur dengan menggunkan PCA terhadap efisiensi waktu komputasi dan akurasi pada metode RGSOM.
5. Mengetahui bagaimana visualisasi peta topografi pada metode RGSOM dapat memberikan pengetahuan baru pada proses pembelajaran.
6. Mengetahui kemampuan metode RGSOM dalam melakukan deteksi intrusi jaringan terkait efisiensi waktu komputasi dan akurasi.

1.4 Manfaat

Hasil dari penulisan tesis ini diharapkan memiliki manfaat sebagai berikut:

1. Menghasilkan nilai parameter yang sesuai pada metode RGSOM agar waktu komputasi dan akurasi dapat optimal pada deteksi intrusi jaringan.
2. Memberikan pengetahuan tentang perbandingan waktu komputasi pada metode RGSOM dalam proses deteksi intrusi jaringan.
3. Mengetahui perbandingan akurasi dari klasifikasi metode RGSOM dengan metode SOM, GSOM, dan SVM sehingga dapat digunakan untuk pengembangan dalam menyelesaikan permasalahan deteksi intrusi di kemudian hari.
4. Memberikan informasi tentang pengaruh reduksi fitur dengan PCA terhadap waktu komputasi dan akurasi pada metode RGSOM.
5. Mengetahui informasi tentang pengetahuan baru yang didapat dari visualisasi peta topografi pada metode RGSOM.
6. Memberikan kontribusi terhadap system deteksi intrusi jaringan dengan metode RGSOM sebagai proses klasifikasi.

1.5 Batasan masalah

Ruang lingkup masalah penelitian pada tesis ini adalah sebagai berikut:

1. Data yang digunakan diambil dari KDD 1999, yang terdiri dari data latih dan data uji.
2. Kelas atau kategori intrusi yang digunakan adalah *Denial-of-Service (DoS), Remote to Local (R2L), User to Root (U2R), Probing*, sehingga terdapat 4 kelas dengan kategori intrusi dan satu kelas dengan label normal.
3. Restricted Growing Self-Organizing Map digunakan sebagai metode deteksi intrusi jaringan dengan cara melakukan klasifikasi.

1.6 Sistematika pembahasan

Untuk mencapai tujuan yang diharapkan, maka sistematika penulisan yang disusun dalam tesis ini adalah sebagai berikut:

BAB 1 Pendahuluan

Bagian pendahuluan terdiri dari latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika pembahasan.

BAB 2 Landasan Kepustakaan

Pada bagian landasan kepustakaan akan diuraikan dasar teori dan kajian pustaka yang berkaitan dengan domain permasalahan, penelitian sebelumnya, tentang metode metode yang digunakan dalam proses klasifikasi dan visualisasi. Dan kemudian landasan penggunaan *Restricted Growing Self-Organizing Map* dalam proses klasifikasi dan visualisasi yang akan di implementasikan pada data KDD 1999.

BAB 3 Metodologi

Metodologi yang digunakan dalam penulisan adalah *Restricted Growing Self-Organizing Map* pada klasifikasi dan visualisasi data KDD 1999. Perancangan sistem meliputi penentuan *data set*, pengolahan data perancangan arsitektur metode *Restricted Growing Self-Organizing Map,* perancangan metode pengujian.

BAB 4 Hasil

Pada bagian ini akan disajikan hasil dari pengujian dari metode *Restricted Growing Self-Organizing Map* dengan menggunakan *data set* KDD 1999, dan juga menyajikan hasil dari metode pembanding.

BAB 5 Pembahasan

Pembahasan meliputi pengujian dari hasil implementasi *Restricted Growing Self-Organizing Map* sebagai metode klasifikasi dan visualisasi analisis data pada KDD 1999.

BAB 6 Penutup

Bagian penutup berisi kesimpulan dari bagian-bagian yang telah diuraikan sebelumnya beserta saran-saran untuk proses pengembangan penelitian lanjutan.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Sistem deteksi intrusi

Perangkat keras atau lunak untuk melakukan deteksi terhadap intrusi jaringan dinamakan *Intrusion Detection System* (IDS). IDS bertujuan untuk melindungi jaringan komputer dari pengguna yang tidak diijinkan. IDS di bangun untuk melakukan deteksi intrusi dengan cara membangun model yang mampu memisahkan antara koneksi yang merupakan aktivitas berbahaya (intrusi atau serangan) dengan koneksi yang normal.

2.1.1 Intrusi

Beberapa kategori intrusi atau akifitas berbahaya pada jaringan komputer adalah sebagai berikut:

1. *Denial-of-Service* (DOS)
2. *Remote to Local* (R2L)
3. *User to Root* (U2R)
4. *Probing*

DOS adalah aktivitas yang menyerang komputer atau server dengan tujuan membuat target tidak mampu menyediakan layanannya untuk pengguna lain dengan cara menghabiskan bandwidth, kapasitas pemrosesan *router* atau *resource* dari jaringan komputer (*network/transport-level flooding attacks*) atau *resource* komputer tersebut (*application-level flooding attacks*) (Zargar, Joshi, & Tipper, 2013). Aktivitas yang dikategorikan sebagai DOS antara lain *syn flood, back, land, neptune, pod, smurf, tear drop*.

R2L adalah aktivitas untuk mencari celah dari suatu komputer atau jaringan target dengan tujuan untuk mendapatkan akses terhadap komputer target atau jaringannya melalui remote komputer. Beberapa jenis dari R2L antara lain *ftp write, guess password, imap, multihop, phf, spy, warezclient, warezmaster.*

U2R merupakan aktivitas dari pengguna normal atau anonym untuk mendapatkan hak akses sebagai *super user* (*root* atau *administrator*) dengan cara melakukan exploitasi terhadap komputer atau jaringan. Beberapa aktivitas yang terkait dengan U2R adalah *buffer overflow, loadmodule, perl, rootkit.*

Probing yaitu salah satu aktivitas berbahaya dalam jaringan komputer yang bertujuan untuk mencari celah kelemahan suatu sistem (komputer maupun jaringan) dengan cara melakukan pemindaian terhadap target. Metode atau teknik (aplikasi) untuk melakukan *probing* biasanya disebut *probe*. Beberapa jenis dari *probe* antara lain adalah *ipsweep, nmap, portsweep, satan.*

2.1.2 Proses deteksi intrusi dengan metode klasifikasi

Untuk dapat membangun sebuah IDS yang efektif dan akurat dalam deteksi terhadap intrusi jaringan dapat dilakukan dengan metode klasifikasi. Proses deteksi intrusi dengan menggunakan metode klasifikasi pada umumnya dilakukan dengan cara membuat model yang merepresentasikan intrusi atau normal dari data latih. Proses pembuatan model ini biasanya disebut dengan proses pembelajaran. Setelah proses pembelajaran selesai, dilakukan proses uji untuk mengetahui tingkat akurasi dari metode klasifikasi tersebut. Proses uji ini menggunakan data uji yang telah diketahui labelnya (intrusi atau normal). Input dari data uji akan diklasifikasikan berdasarkan model yang telah dibuat pada proses pembelajaran. Hasil klasifikasi akan dibandingkan dengan label sebenarnya pada data uji.

Hasil klasifikasi dinyatakan sebagai *True Positif* (TP) apabila hasil klasifikasi sesuai dengan label aslinya dan berjenis intrusi. Bernilai *True Negatif* (TN) jika hasil klasifikasi sesuai dengan label aslinya tetapi berkategori normal. Sedangkan apabila hasil klasifikasi diberi label intrusi tetapi tidak sesuai dengan label aslinya maka disebut sebagai *False Positif* (FP) karena label sebenarnya adalah normal. Hasil klasifikasi dinyatakan bernilai *False Negatif* (FN) ketika hasil klasifikasi tidak sesuai dengan label aslinya dan diberi label normal tetapi sesungguhnya adalah intrusi. Hasil klasifikasi ini akan digunakan untuk melakukan pengukuran akurasi yang akan dijelaskan pada sub bab 2.9.

Proses pembelajaran dan pelabelan atau pengklasifikasian pada tiap metode klasifikasi secara khusus memiliki cara yang berbeda beda antar metode. Langkah langkah pada metode klasifikasi akan dijelaskan pada sub bab tentang metode tersebut. Untuk menghasilkan model pada proses pembelajaran dan menghasilkan hasil klasifikasi yang akurat diperlukan data latih dan data uji yang dapat mencakup beberapa intrusi atau aktivitas berbahaya. Pada penelitian tesis ini dipergunakan data yang diambil dari KDD 1999 yang memiliki jumlah yang besar dan yang terutama dapat merepresentasikan berbagai macam intrusi yang telah disebutkan diatas.

2.2 Data KDD 1999

Kumpulan data KDD 1999 diambil dari repositori University of California Irvine (http://archive.ics.uci.edu/ml/datasets/kdd+cup+1999+data) yang terdiri dari 10% data latih (494.021 *instance*) dan data uji (4.898.431 *instance*) dengan 41 atribut atau fitur. Kategori *data set* ini terbagi dalam 4 kelas intrusi dos, probe, r2l, u2r dan satu kategori dengan label normal. Jumlah sebaran distribusi kategori yang terdapat pada data latih dapat dilihat pada Tabel 2.1.

Dari penelitian terkait IDS yang mempergunakan data dari KDD 1999, tidak semuanya mempergunakan data latih dan uji yang disediakan. Tetapi, penelitian ini akan mempergunakan keseluruhan data yang disediakan. Semua data tersebut digunakan untuk mengukur efisiensi waktu komputasi dan akurasi dari metode yang diusulkan dalam permasalahan deteksi intrusi dengan metode klasifikasi.

Tabel 2.1 Distribusi jumlah kategori data latih dan data uji pada KDD 1999

Intrusi	Data Latih	Data Uji
Dos	391458	3883370
Normal	97278	972781
Probe	4107	41102
R2L	1126	1126
U2R	52	52
Total	**494021**	**4898431**

Dari sebaran jumlah kategori pada data latih yang di tunjukkan pada Tabel 2.1 kategori dari kumpulan data ini termasuk data *imbalanced* atau tidak seimbang. Data itu dikatakan tidak seimbang karena jumlah kelas DDoS jumlahnya sangat besar dibandingkan dengan kelas yang lainnya. Dalam penentuan akurasi terhadap data yang tidak berimbang ini diperlukan ukuran lainya seperti *presision, recall,* dan *false alram rate*. Pengukuran akurasi akan di jelaskan lebih detail pada sub bab 2.9.

Berdasarkan KDD 1999 Task Description (1999) yang diadaptasi dari S. J. Stolfo et al. (2000), melakukan pembagian fitur yang terdapat pada *data set* KDD 1999 berdasarkan tiga kategori, yaitu :

- Fitur dasar dari koneksi TCP secara individu (Tabel 2.2)
- Fitur konten dari koneksi berdasarkan domain knowledge (Tabel 2.3)
- Fitur traffic yang dihitung menggunakan *two-second time window* (Tabel 2.4)

Tabel 2.2 Fitur dasar dari koneksi TCP secara individu

Nama Fitur	Deskripsi	Tipe
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc	discreate
service	network service on the destination, e.g., http, telnet	discreate
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of ``wrong'' fragments	continuous
urgent	fragment number of ``wrong'' fragments continuous urgent number of urgent packets	continuous

Sumber: KDD 1999 Task Description (1999)

Tabel 2.3 Fitur konten dari koneksi berdasarkan *domain knowledge*

Nama Fitur	Deskripsi	Tipe
hot	number of ``hot'' indicators	continuous
num_failed_logins	number of failed login attempts	continuous
logged_in	1 if successfully logged in; 0 otherwise	discreate
num_compromised	number of ``compromised'' conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if ``su root'' command attempted; 0 otherwise	discrete
num_root	of ``root'' accesses	continuous
num_file_creation	number of file creation operations	
num_shell	number of shell prompts continuous	
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_hot_login	if the login belongs to the ``hot'' list; 0 otherwise	discrete
is_guest_login	1 if the login is a ``guest''login; 0 otherwise	discrete

Sumber: KDD 1999 Task Description (1999)

Tabel 2.4 Fitur *traffic* yang dihitung menggunakan *two-second time window*

Nama Fitur	Deskripsi	Tipe
count	number of connections to the same host as the current connection in the past two seconds. Note: The following features refer to these same host connections.	continuous
serror_rate	% of connections that have ``SYN'' errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds. Note: The following features refer to these same service connections	continuous
srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous

Sumber: KDD 1999 Task Description (1999)

2.3 Principle Component Analysis sebagai metode reduksi fitur

PCA berfungsi untuk mengukur dimensi dan mengidentifikasi titik-titik data dengan kemungkinan varian tertinggi. Hasil PCA untuk memproyeksikan ruang di subruang yang lebih kecil yang mewakili data dengan mengurangi dimensi fitur. Proyeksi ke ruang yang lebih kecil memungkinakan untuk melakukan transformasi ke dimensi yang lebih rendah (reduksi fitur), hal ini dapat mengurangi waktu komputasi dan kesalahan estimasi parameter (Ikram, 2016).

Pendekatan PCA standar dapat dirangkum dalam enam langkah sederhana:

a. Hitung matriks kovarian dari *data set* d-dimensi ternormalisasi.
b. Hitung vector eigen dan nilai eigen dari matriks kovarian.
c. Urutkan nilai eigen dalam urutan menurun.
d. Pilih vektor eigen yang sesuai dengan nilai eigen terbesar di mana k adalah jumlah dimensi dari subruang fitur baru.
e. Bangun matriks proyeksi dari vector eigen terpilih.
f. Transformasi *data set* asli untuk membangun ruang fitur k-dimensi baru.

2.4 Support Vector Machine

SVM merupakan salah satu pengklasifikasi berdasarakan fungsi diskriminan linier, yang ideal digunakan untuk klasifikasi biner dan bisa digunakan juga untuk melakukan klasifikasi dengan banyak kelas (*multiclass*) (Murty & Raghava, 2016).

$$\boldsymbol{W}^t\boldsymbol{X} + \boldsymbol{b}$$

Beberapa hal yang penting pada SVM antara lain (Murty & Raghava, 2016):

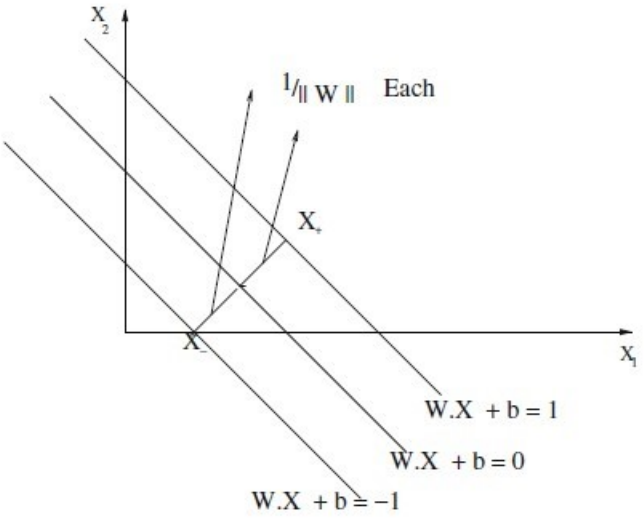
1. Memaksimalkan jarak antar kelas berdasarkan permasalahan. Pada kasus yang dapat dipisahkan secara linier, sangat memungkinkan untuk memperoleh global optimum dari $\boldsymbol{W}$.
2. Dapat mempelajari *nonlinier boundaries* pada input dengan melakukan pemetaan dari input *space* ke fitur berdimensi tinggi, dan kemudian belajar dari fitur *space*.

2.4.1 Linier separability

Support planes adalah *hyper plane* yang memenuhi persamaan $\boldsymbol{W}^t\boldsymbol{X}_i + b = 1$ dan $\boldsymbol{W}^t\boldsymbol{X}_i + b = -1$. Vektor pada data latih yang terletak pada *support plane* dapat menjadi *support vector*. Ketika kelas merupakan *linearly separable* (dapat dipisahkan secara linier), kita dapat melakukan penskalaan $\boldsymbol{W}$ dan b untuk membentuk *support planes* agar memenuhi kaidah persamaan $\boldsymbol{W}^t\boldsymbol{X}_i + b = 1$ dan $\boldsymbol{W}^t\boldsymbol{X}_i + b = -1$.

2.4.2 Margin

Jarak antara dua *plane* disebut dengan margin. Margin adalah fungsi dari ***W.*** Pembelajaran pada SVM merupakan proses pembelajaran dimana memperoleh nilai margin yang maksimal dari fungsi ***W***.



Gambar 2.1 Margin dari SVM

Sumber: Murty dan Raghava (2016)

Memperhatikan Gambar 2.1. Terdapat tiga buah garis parallel pada bidang dua dimensi dimana $\boldsymbol{W}.\boldsymbol{X}$ adalah *dot product* dan sama dengan $\boldsymbol{W^tX}$. Garis garis tersebut merepresentasikan:

1. $\boldsymbol{W}.\boldsymbol{X} + b = -1$ adalah garis support *plane* dari kelas negatif.
2. $\boldsymbol{W}.\boldsymbol{X} + b = 0$ adalah *decision boundary* antara dua kelas.
3. $\boldsymbol{W}.\boldsymbol{X} + b = 1$ adalah *support plane* dari kelas positif.

Jarak dari $\boldsymbol{X}_+$ ke hyperplane $\boldsymbol{W}.\boldsymbol{X} + b = 0$ adalah $d = \frac{g(\boldsymbol{X}_+)}{\|\boldsymbol{W}\|}$; karena $\boldsymbol{X}_+$ berada pada garis $g(x) = \boldsymbol{W}.\boldsymbol{X} + b = 1$, maka $g(\boldsymbol{X}_+) = 1$, sehingga $d = \frac{1}{\|\boldsymbol{W}\|}$.

Demikian pula jarak $\boldsymbol{X}_-$ ke hyperplane $\boldsymbol{W}.\boldsymbol{X} + b = 0$ adalah $d = \frac{g(\boldsymbol{X}_-)}{\|\boldsymbol{W}\|}$, maka jarak $d = \frac{1}{\|\boldsymbol{W}\|}$. Jadi,

$$Margin = \frac{1}{\|\boldsymbol{W}\|} + \frac{1}{\|\boldsymbol{W}\|} = \frac{2}{\|\boldsymbol{W}\|} = \frac{2}{\boldsymbol{W^t W}} \quad (2.1)$$

Untuk mendapatkan nilai ***W,*** dari pada melakukan maksimasi nilai $\frac{2}{\boldsymbol{W^t W}}$, dapat dilakukan dengan melakukan minimasi terhadap nilai $\frac{\boldsymbol{W^t W}}{2}$. Dengan menggunakan Lagrangian dengan tujuan untuk merubah permasalahan *constrained optimization* menjadi permasalahan *unconstrained optimization* berdasarkan persamaan 2.2.

$$\mathcal{L} = \frac{1}{2}\boldsymbol{W^t W} + \sum_{i=1}^{n} \alpha_i (1 - y_i(\boldsymbol{W^t X_i} + b)) \quad (2.2)$$

Dan mencari nilai optimum dengan cara mencari gradient terhadap $\boldsymbol{W}$ sesuai dengan persamaan 2.3 dan terhadap *b* berdasarkan persamaan 2.4.

$$\frac{\delta \mathcal{L}}{\delta \boldsymbol{W}} \Rightarrow \boldsymbol{W} + \sum_{i=1}^{n} \alpha_i (-y_i) \boldsymbol{X_i} = 0$$

$$\Rightarrow \boldsymbol{W} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{X_i} \quad (2.3)$$

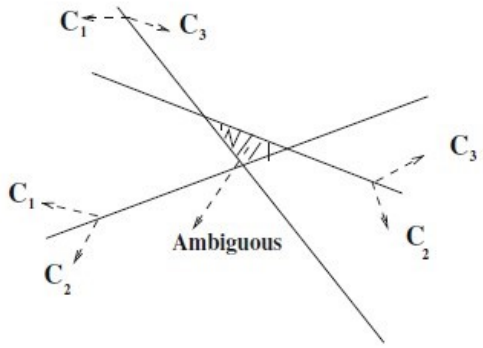
$$\frac{\delta \mathcal{L}}{\delta b} \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0 \quad (2.4)$$

2.4.3 Permasalahan multi kelas

Berdasarkan Wang & Xue (2014), untuk menyelesaikan permasalahan lebih dari 2 kelas terdapat dua buah pendekatan antara lain secara tidak langsung (*Indirect Multi-Class SVM*) dan langsung (*Direct Multi-Class SVM*). Pada pendekatan secara tidak langsung ada dua metode dasar yaitu pendekatan One-Versus Rest (1VR) dan One-Versus-One (1V1). Pada pendekatan secara langsung terdapat metode *Weston and Watkins' Multi-Class SVM*, *Vapnik Multi-Class SVM*, *Crammer and Singer's Multi-Class SVM*, dan *Simplified Multi-Class SVM*.

Pendekatan *One Versus Rest* membangun sejumlah *k* pengklasifikasi biner secara terpisah, untuk data dengan jumlah kelas *k*. Pengklasifikasi biner ke-*m* dilatih menggunakan data dari kelas ke-*m* sebagai contoh positif dan sisa kelas *k-1* sebagai contoh negatif. Selama pengujian, label kelas ditentukan oleh pengklasifikasi biner yang memberikan nilai output maksimum. Pendekatan *One-Versus-One* atau yang disebut dengan *pairwise decomposition* menggunakan langkah sebagai berikut:

1. Membuat pasangan atar kelas, dengan jumlah pasangan dihitung berdasarkan persamaan $\frac{C(C-1)}{2}$, dengan setiap pasangan menggunakan linier SVM.
2. Komplementari region untuk kelas $\bar{C}_i$ adalah sebagai berikut:
3. $\bar{C}_i = \bigcup_{j=1, j \neq i}^{C} C_j$Penentuan label dengan cara mencari jumlah vote terbanyak dari setiap pasangan.



Gambar 2.2 Klasifikasi multi kelas

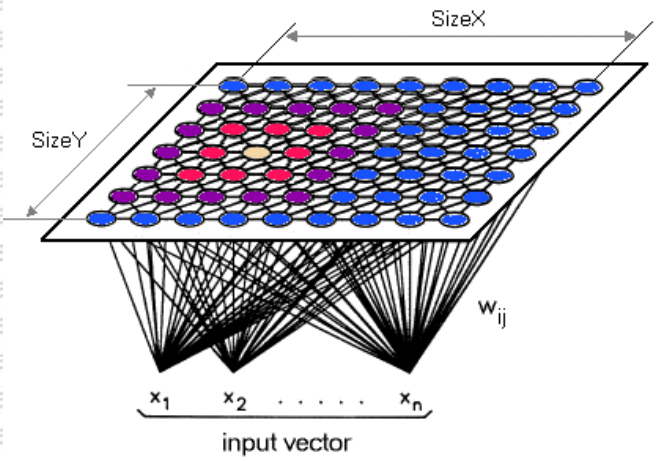
Sumber: Murty dan Raghava (2016)

Pada pendekatan secara langsung, proses klasifikasi dilakukan secara langsung atas keseluruhan kelas, bukan dengan cara membuat beberapa pengklasifikasi biner. Untuk permasalahan kelas sejumlah *k*, metode ini mendesain fungsi obyektif tunggal untuk melatih pengklasifikasi biner sejumlah *k* secara bersamaan dan memaksimalkan margin dari masing-masing kelas.

2.5 Self-Organizing Map

Self Organizing Map (SOM) (Kohonen, 2013) merupakan *nonlinearly projecting mapping* dengan metode pembelajaran *unsupervised* dan juga bisa menggunakan metode pembelajaran *supervised*. Tujuan dari SOM adalah melakukan reduksi data dari data berdimensi tinggi menjadi berdimensi rendah (2-3 dimensi) yang sangat berguna untuk visualisasi data. SOM selain melakukan reduksi data berdimensi tinggi juga mampu melakukan klasifikasi.

Model SOM biasanya diasosiasikan dengan node dari grid dua dimensi seperti pada Gambar 2.3, yang biasa disebut dengan vektor referensi atau peta topografi. Algoritma SOM membentuk model dengan aturan bahwa semakin memiliki kesamaan maka akan diasosiasikan dengan node yang terdekat pada vektor referensi. Ide dari SOM adalah setiap data input memilih model yang terdekat dengan input yang dinamakan dengan neuron/node pemenang (*winner*), dan node tetangga dari *winner* harus di modifikasi agar nilainya lebih mendekati input.



Gambar 2.3 Arsitektur Self-Organizing Map

Sumber: www.lohninger.com (2012)

Penentuan *winner* dilakukan dengan menggunakan persamaan 2.5. Dimana $\boldsymbol{X}$ adalah Input dan $\boldsymbol{W}_j$ adalah neuron/node ke *j* pada vektor referensi, dan *c* adalah index dari *winner* dimana model $\boldsymbol{W}_c$ merupakan model yang memiliki jarak terdekat dengan input $\boldsymbol{X}$.

$$c = \operatorname{argmin}\{\| \boldsymbol{X}(t) - \boldsymbol{W}_j\,(t) \|\} \quad (2.5)$$

Pada SOM reguler untuk menentukan modifikasi dari vektor referensi dilakukan dengan menggunakan persamaan 2.6, dimana h_{ci} merupakan *neighborhood function*.

$$\boldsymbol{W}_j(t+1) = \boldsymbol{W}_j(t) + h_{ci}(\mathbf{X}(\mathrm{t}) - \boldsymbol{W}_j(t)) \quad (2.6)$$

Metode SOM reguler ini memiliki kelemahan jika implementasikan pada data berukuran besar. Kelemahan tersebut terkait besarnya ukuran vektor referensi, karena pada metode SOM reguler setiap input akan menempati sebuah node pada peta topografi. Semakin besar jumlah vektor referensi maka semakin bertambah waktu yang dibutuhkan untuk menentukan *winner* pada proses pembelajaran dan pengujian.

2.6 Growing SOM

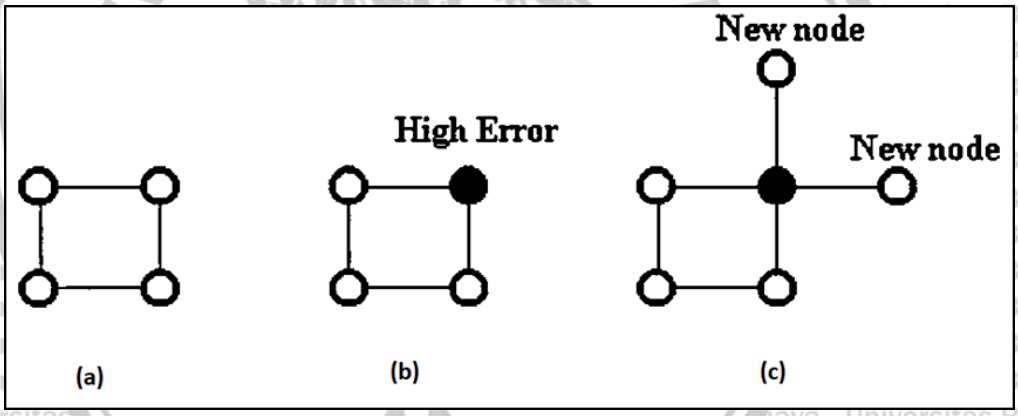
Growing SOM (GSOM) adalah modifikasi SOM reguler yang ajukan oleh Alahakoon (2000). Prosedur GSOM dibagi menjadi tiga fase yaitu *Initialization*, *Growing,* dan *Smoothing*.

1. *Initialization Phase*:
   a) Inisialisasi bobot vektor referensi dari node awal (biasanya berjumlah empat) dengan nilai acak, Gambar 2.4 (a).
   b) Menghitung *growth threshold* (GT) berdasarkan kebutuhan (persamaan 2.7).
2. *Growing Phase:*
   a) Memproses input ke jaringan.

b) Menentukan node vektor referensi yang terdekat dengan input sebagai *winner*.
c) Melakukan update bobot vektor referensi yang merupakan tetangga (*neighbors*) dari *winner* dan *winner* itu sediri.
d) Menaikkan nilai error dari *winner*.
e) Ketika $TE_i > GT$ (dimana TE_i adalah total error dari node ke i dan GT adalah *growth threshold*), Gambar 2.4(b). Tambah node jika merupakan *boundary node*. Distribusikan bobot ke tetangga jika bukan *boundary node*.
f) Inisialisasi node yang baru dengan bobot yang sesuai dengan bobot tetangga, Gambar 2.4(c).
g) Inisialisasi *learning rate* (LR) ke nilai awal.
h) Ulangi langkah b) - g) sampai semua input ditempatkan dan pertumbuhan node menurun sampai ke level minimum.

3. *Smoothing Phase:*

a) Turunkan *learning rate* dan betulkan *neighborhood* awal yang kecil.
b) Cari *winner* dan update bobot dari *winner* dan tetangga dengan cara yang sama seperti fase pertumbuhan.



Gambar 2.4 Proses generasi node pada GSOM.
(a) Initial node berjumlah 4, (b) Winner node yang memiliki nilai error melebihi batas, (c) Proses generasi node baru.

Sumber : Alahakoon (2000)

$$GT = -D \times \ln(SF) \quad (2.7)$$

Dimana GT adalah *growth threshold* dan *D* adalah dimensi data, sedangkan *SF* adalah *spread factor*.

Metode GSOM hanya mampu mengatasi permasalahan efisiensi pada epoch pertama yaitu pada *growing phase*. Pada *smoothing phase*, jumlah vektor referensi sudah semakin banyak karena tidak dibatasi jumlahnya sehingga permasalahan efisiensi akan muncul kembali. Permasalahan ini yang akan diselesaikan menggunakan metode RGSOM.

2.7 Normalisasi

Fitur-fitur yang terdapat pada *data set* harus diolah terlebih dahulu dengan cara dilakukan normalisasi yang bertujuan untuk melakukan penskalaan pada data sehingga mempunyai jarak (*range*) yang sama antar fitur sehingga mengurangi bias. Menurut Jayalakshmi dan Santhakumaran (2011), normalisasi juga berguna untuk mempercepat proses pembelajaran. Pada penelitian ini normalisasi menggunakan persamaan 2.8, dimana x adalah nilai awal dan x' adalah nilai setelah dinormalisasi, sedangkan F_i adalah fitur ke-*I,* $argmin(F_i)$ adalah nilai minimum dari fitur ke-*i,* sedangkan $argmax\ (F_i)$ adalah nilai maksimal dari fitur ke-*i*.

$$x' = \frac{x - argmin(F_i)}{argmax\ (F_i) - argmin(F_i)} \quad (2.8)$$

2.8 Inisialisasi vektor veferensi (*initial map*)

Cara inisialisasi vector referensi atau *initial map* pada metode SOM dan turunannya terbagi menjadi dua acara, yaitu *Random Initialization* (RI) dan *PCA initialization* (PCI) (Akinduko, Mirkes, & Gorban, 2016). Pada RI inisialisasi bobot vector referensi dilakukan secara acak mengacu pada range normalisasi data latih. Pada PCI, inisialisasi bobot vector referensi di pilih berdasarkan kombinasi linier dari dua principle component pertama yang terbentuk pada proses PCA terhadap data latih. PCI di lakukan dengan tujuan agar proses pembobotan awal mendekati kondisi bobot akhir dari vector referensi yang berdampak pada waktu komputasi yang lebih cepat (Kohonen, 2013).

Sebagai contoh untuk membentuk initial map dengan ukuran 5x3, maka terlebih dahulu di tentukan factor pengali dengan ukuran array 5 dan 3, dimana factor pengali adalah a1 = [-1.0 -0.5 0.0 0.5 1.0] dan a2 = [-1.0 0.0 1.0]. Bobot dari initial map dihitung berdasarkan persamaan 2.9:

$$\mathrm{w[m,n]} = \mathrm{a1[m]} * \mathrm{e1} + \mathrm{a2[n]} * \mathrm{e2} \quad (2.9)$$

, dimana m= 1, 2, ... ,5 (jumlah kolom) dan n = 1,2,3 (jumlah baris).

2.9 Metode pengukuran akurasi

Pada tesis ini akan digunakan empat metode pengukuran untuk melakukan evaluasi pada metode yang akan diajukan. Metode pengukuran tersebut antara lain: accuracy (ACC), *false alarm rate* (FAR), *detection rate* (DTR) atau *recall*, *precision*. Akurasi adalah total klasifikasi yang benar dibandingkan dengan jumlah total data test. Akurasi dihitung berdasarkan Persamaan 2.10:

$$\mathrm{ACC} = \frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FP+FN}} \quad (2.10)$$

False alarm rate adalah jumlah kategori normal yang diberi label sebagai intrusi dibandingkan dengan jumlah total dari data test, sesuai dengan persamaan 2.11:

$$\mathrm{FAR} = \frac{\mathrm{FP}}{\mathrm{TN+FP}} \quad (2.11)$$

Detection rate atau *recall* adalah jumlah intrusi yang diberi label dengan benar dibandingkan dengan jumlah total intrusi pada data test sesuai dengan persamaan 2.12.

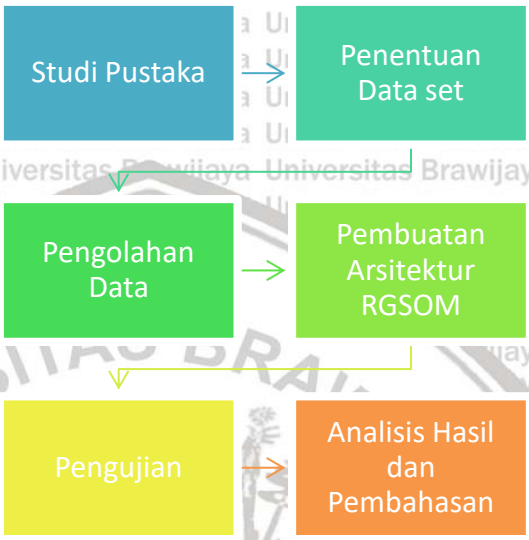
$$\text{DTR or Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (2.12)$$

Precision merupakan jumlah intrusi yang diberi label secara benar dibanding jumlah total dari data test yang diberi label sebagai intrusi, yang di hitung berdasarakan persamaan 2.13.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (2.13)$$

BAB 3 METODOLOGI

Pada bab berikut ini akan membahas tentang metode penelitian yang akan digunakan dalam penelitian yang berjudul Sistem Deteksi Intrusi dengan Metode Restricted Growing Self-Organizing Map. Gambar 3.1 adalah diagram alur metode penelitian yang akan digunakan dalam penelitian.



Gambar 3.1 Diagram alur metode penelitian

3.1 Studi pustaka

Studi pustaka yang digunakan oleh penulis berasal dari buku, publikasi ilmiah dari jurnal internasional dan dari situs internet yang memuat informasi terkait penelitian. Studi pustaka yang akan diambil mencakup kategori intrusi apa saja yang diakibatkan oleh serangan siber, metode yang telah digunakan pada penelitian sebelumnya untuk melakukan deteksi terhadap intrusi. Metode yang biasa digunakan terkait dengan rumusan permasalahan nomor 1 tentang, bagaimana bagaimana klasifikasi dan visualisasi intrusi berdasarkan *data set* KDD 1999. Metode yang terkait dengan self-organizing map, dan support vektor machine akan mendapat prioritas karena terkait dengan metode yang akan di ajukan dan bagian dari metode pembanding.

3.2 Penentuan data set

Berdasar studi pustaka yang telah dilakukan terdapat beberapa *data set* yang biasa digunakan sebagai pengujian deteksi intrusi, antara lain DARPA, KDD 1999, dan NSL-KDD *Data set*. Pada penelitian ini akan digunakan KDD 1999 sebagai *data set* yang akan digunakan sebagai *benchmark* terhadap metode yang diusulkan dibandingkan dengan metode SVM. KDD 1999 diambil dari repositori UCI yang terdiri dari 494.021 instance data latih dan 4.898.431 instance data uji dan memiliki 41 atribut (fitur).

Jumlah data yang besar ini bertujuan untuk menguji efisiensi, akurasi, *false alarm rate, detection rate atau recall, precision* dari metode yang diajukan untuk mengatasi permasalahan data berukuran besar pada pendeteksian intrusi. Karakteristik dari data KDD 1999 memiliki kesesuaian dalam membuat model deteksi intrusi berbasis anomali, sehingga kumpulan data ini sesuai jika digunakan untuk menguji apakah metode yang akan digunakan nanti mampu melakukan deteksi intrusi secara adaptif dan fleksibel.

3.3 Pengolahan data (*Preprocessing*)

Sebelum dilakukan proses pembelajaran dan pengklasifikasian, data latih maupun data uji harus diolah terlebih dahulu pada tahap *preprocessing*. Pengolahan data tersebut antara lain adalah proses *label encoder*, normalisasi atau penskalaan data, pemetaan kelas dan reduksi fitur (*feature reduction*).

3.3.1 Label encoder

Label encoder merupakan proses untuk merubah data yang berupa *string* menjadi angka. Pada data KDD 1999 ada tiga fitur yang dilakukan proses *label encoder*, antara lain *protocol_tipe, flag,* dan *service.* Tabel 3.1 menunjukkan hasil *label encoder* nilai fitur dari *protocol_type*. Sedangkan hasil label encoder dari fitur *flag* ditunjukkan pada Tabel 3.2. *Label encoder* nilai fitur *service* ditunjukkan pada Tabel 3.3.

Tabel 3.1 Label encoder nilai fitur *protocol_type*

No	protocol_type	nilai x
1	icmp	0
2	tcp	1
3	udp	2

Tabel 3.2 Label encoder nilai fitur *flag*

No	flag	nilai x
1	OTH	0
2	REJ	1
3	RSTO	2
4	RSTOS0	3
5	RSTR	4
6	S0	5
7	S1	6
8	S2	7
9	S3	8
10	SF	9
11	SH	10

Tabel 3.3 Label encoder nilai fitur service

No	service	nilai
1	IRC	0
2	X11	1
3	Z39_50	2
4	aol	3
5	auth	4
6	bgp	5
7	courier	6
8	csnet_ns	7
9	ctf	8
10	daytime	9
11	discard	10
12	domain	11
13	domain_u	12
14	echo	13
15	eco_i	14
16	ecr_i	15
17	efs	16
18	exec	17
19	finger	18
20	ftp	19
21	ftp_data	20
22	gopher	21
23	harvest	22
24	hostnames	23

No	service	nilai
25	http	24
26	http_2784	25
27	http_443	26
28	http_8001	27
29	imap4	28
30	iso_tsap	29
31	klogin	30
32	kshell	31
33	ldap	32
34	link	33
35	login	34
36	mtp	35
37	name	36
38	netbios_dgm	37
39	netbios_ns	38
40	netbios_ssn	39
41	netstat	40
42	nnsp	41
43	nntp	42
44	ntp_u	43
45	other	44
46	pm_dump	45
47	pop_2	46
48	pop_3	47

No	service	nilai
49	printer	48
50	private	49
51	red_i	50
52	remote_job	51
53	rje	52
54	shell	53
55	smtp	54
56	sql_net	55
57	ssh	56
58	sunrpc	57
59	supdup	58
60	systat	59
61	telnet	60
62	tftp_u	61
63	tim_i	62
64	time	63
65	urh_i	64
66	urp_i	65
67	uucp	66
68	uucp_path	67
69	vmnet	68
70	whois	69

3.3.2 Normalisasi

Normalisasi merupakan proses yang bertujuan untuk penskalaan nilai yang berfungsi mempercepat proses pembelajaran dan pengklasifikasian. Pada penelitian tesis ini normalisasi menggunakan persamaan 3.1, sesuai dengan persamaan 2.8. Normalisasi ini disebut dengan *min max scaler,* nilai akan dikonversi menjadi antara 0 sampai 1 dengan nilai minimum akan menjadi 0 dan nilai maksimum akan menjadi 1.

$$x' = \frac{x - argmin(F_i)}{argmax\ (F_i) - argmin(F_i)} \quad (3.1)$$

3.3.3 Pemetaan kelas

Sub kelas yang terdapat pada *data set* berjumlah 23 dan dipetakan berdasarkan kelas utama menjadi 5 kelas berdasarkan Tabel 3.4.

Tabel 3.4 Pemetaan kelas

Sub Kelas	Kelas
back.	dos
land.	dos
neptune.	dos
pod.	dos
smurf.	dos
teardrop.	dos
buffer_overflow.	u2r
loadmodule.	u2r

Sub Kelas	Kelas
perl.	u2r
rootkit.	u2r
ftp_write.	r2l
guess_passwd.	r2l
imap.	r2l
multihop.	r2l
phf.	r2l
spy.	r2l

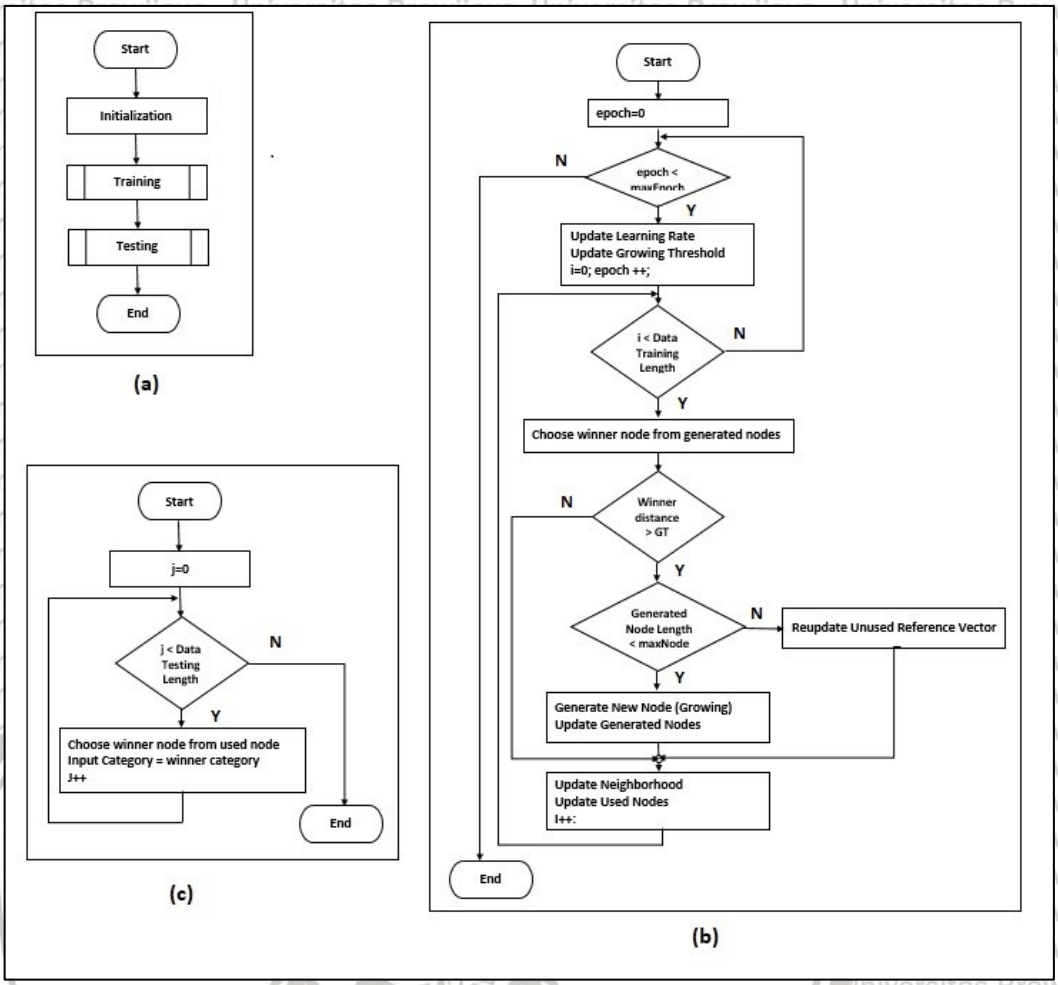
Sub Kelas	Kelas
warezclient.	r2l
warezmaster.	r2l
ipsweep.	probe
nmap.	probe
portsweep.	probe
satan.	probe
normal.	normal

3.3.4 Reduksi fitur (*feature reduction*)

Setelah data diolah menggunakan *label encoder*, normalisasi, dan dilakukan pemetaaan kelas langkah selanjutnya adalah mereduksi fitur yang berjumlah 41, menjadi beberapa fitur saja berdasarkan metode *Principal Component Analysis* (PCA). PCA akan digunakan untuk mereduksi fitur dengan persyaratan terpenuhinya 95% varian yang akan membentuk *Pricipal Component*-nya. Fitur yang semula (41 fitur) akan direduksi menjadi sejumlah *principal component* yang terbentuk. Nilai pada fitur akan di transformasikan berdasarkan *principal component* yang terbentuk untuk menjadi fitur baru sejumlah principle component yang terbentuk.

3.4 Restricted Growing SOM (RGSOM)

RGSOM adalah pengembangan dari SOM dan GSOM dengan kemampuan untuk melakukan penambahan jumlah node (neuron) apabila jarak *winner* melebihi dari *Growing Threshold* (GT) dan dibatasi oleh jumlah node maksimum atau *map size* (MS) yang bisa di bentuk pada peta topograpi (Christyawan et al., 2019). Penggunaan metode *Clustering Reference Vector* (CRV) pada RGSOM bertujuan agar satu node vektor referensi dapat memuat lebih dari satu input dengan batasan *clustering threshold* (CT). Sebuah input dapat menempati suatu node pada vektor referensi apabila jarak dengan *winner* kurang dari nilai CT. Nilai CT bisa sama dengan nilai GT atau bisa berbeda, dalam penelitian ini nilai CT sama dengan nilai GT.



Gambar 3.2 Diagram alir prosedur RGSOM

Sumber : (Christyawan et al., 2019)

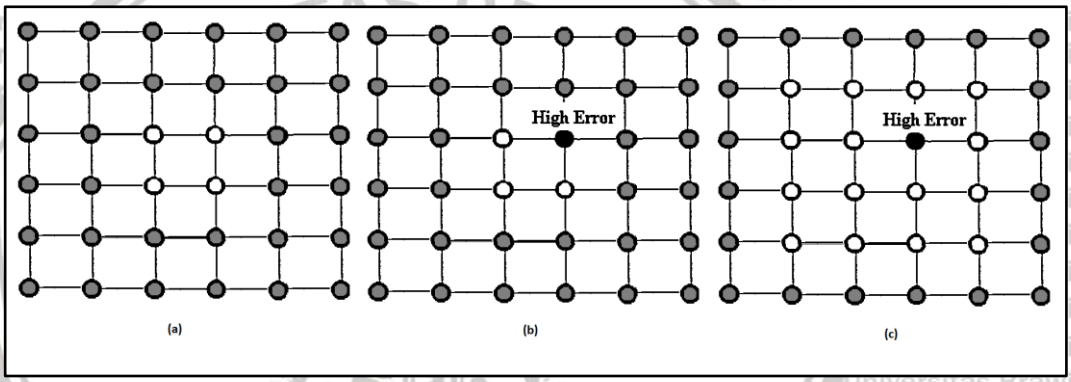
(a) Diagram alir keseluruhan, (b) Prosedur pembelajaran pada RGSOM, (c) Prosedur klasifikasi data uji pada RGSOM.

Prosedur RGSOM dapat dilihat pada Gambar 3.2 (a). Tahap pertama merupakan proses inisialisasi parameter dan inisialisasi bobot vektor referensi. Tahap pertama ini hampir sama dengan initialization phase pada GSOM, yang menjadi perbedaannya adalah parameter-parameternya. Tahap kedua adalah proses pembelajaran (*training/learning*) dari RGSOM. Tahap ketiga merupakan proses klasifikasi (*testing*) yang menggunakan data uji dari KDD 1999.

Diagram alir prosedur tahap pembelajaran ditunjukkan pada Gambar 3.2 (b), tetapi pada penelitian ini melakukan modifikasi pada proses pembelajaran RGSOM. Modifikasi langkah-langkah pada tahap pembelajaran adalah sebagai berikut:

a) Update *learning rate* dan *growing threshold* berdasarkan epoch.
b) Memproses data latih sebagai input ke jaringan.

c) Menentukan *winner* pada vektor referensi yang terdekat dengan input dan memberikan label pada *winner* sesuai dengan kelas pada input jika belum memiliki label. Node pada vektor referensi yang sudah pernah sebagai *winner* dapat dipilih lagi sebagai *winner* apabila kelas input sama dengan kelas sebelumnya (*clustering reference vector*).
d) Jika D > CT dan URV < MR (dimana D adalah jarak *winner* dan CT adalah *clustering threshold*, URV (*Unused Reference Vector*) adalah jumlah vektor referensi yang tidak terpilih, dan MR adalah radius maksimum, maka dilakukan proses update restricted node (*release* node baru yang sebelumnya masuk pada *restricted area*) jika terdapat node pada restricted area, jika tidak maka akan di tambahkan *flag overload* sebagai penanda bahwa node yang disediakan pada peta topografi kurang dari kebutuhan.
e) Melakukan update bobot vektor referensi yang merupakan tetangga (*neighbors*) dari *winner* dan *winner* itu sediri.
f) Ulangi langkah b) - e) sampai semua input di proses.
g) Ulangi langkah a) - f) sampai epoch maksimum.



Gambar 3.3 Proses pertumbuhan pada RGSOM

(a) Initial node dengan restricted area, (b) *Winner* node yang memiliki nilai error melebihi batas, (c) Proses update restricted area.

Gambar 3.2 (c) merupakan diagram alir proses klasifikasi dengan menggunakan data uji pada RGSOM. Proses klasifikasi ini akan menentukan apakah input akan dinyatakan sebagai TP, TN, FP, atau FN. Jumlah nilai TP, TN, FP, atau FN akan digunakan untuk menentukan akurasi sesuai dengan yang dijelaskan pada Bab 2 tentang metode pengukuran akurasi. Langkah-langkah pada tahap pengujian adalah sebagai berikut:

a) Memproses data uji sebagai input ke jaringan.
b) Menentukan *winner* berdasarkan model vektor referensi yang dihasilkan pada tahap pembelajaran.
c) Input diberi label berdasarkan kelas yang terdapat pada *winner.*
d) Label pada input akan di bandingkan dengan label real data uji untuk menentukan apakah proses klasifikasi adalah TP, TN, FP, atau FN

Perhitungan matematis untuk menentukan *winner* sama dengan SOM reguler menggunakan persamaan 2.5. Perhitungan *Average Squared Error* dinyatakan dengan persamaan 3.2, dimana $h_{win\ j}$ adalah fungsi Gaussian neighborhood dari *winner*.

$$E\ (t) = \frac{1}{2} \sum_j h_{win\ j} \left\| \boldsymbol{x} - \boldsymbol{w}_j \right\|^2 (t) \tag{3.2}$$

Dari *error backpropagation* dan *gradient descen*t untuk melakukan modifikasi pada vektor referensi dinyatakan dengan persamaan 3.3.

$$\boldsymbol{w}_j(t+1) = \boldsymbol{w}_j(t) - \frac{\partial\, E(t)}{\partial\, w_j(t)} \tag{3.3}$$

$$\frac{\partial\, E(t)}{\partial\, \boldsymbol{w}_j(t)} = \frac{\partial \frac{1}{2} \sum_j h_{winj} \|\mathbf{x} - \mathbf{w}_j\|^2 (t)}{\partial\, \boldsymbol{w}_j(t)} \tag{3.4}$$

$$\frac{\partial\, E(t)}{\partial\, \boldsymbol{w}_j(t)} = -\, h_{win\ j} \left\|\mathbf{x} - \mathbf{w}_j\right\| (t) \tag{3.5}$$

$$\boldsymbol{w}_j(t+1) = \boldsymbol{w}_j(t) + h_{win\ j}(\mathbf{x}(\mathrm{t}) - \boldsymbol{w}_j(t)) \tag{3.6}$$

Pada persamaan (3.6) dapat diketahui modifikasi dari vektor referensi. G*aussian neighborhood function* dinyatakan dengan persamaan 3.7, dimana *learning rate* $\alpha(t)$ and $\sigma(t)$ adalah jarak dari *neighborhood radius*.

$$h_{win\ j} = \alpha(t)\, e^{-\left(\frac{\left\|win - r_j\right\|^2}{2\sigma^2\,(t)}\right)} \tag{3.7}$$

Nilai l*earning rate* akan menurun berdasarkan fungsi *monotonically decreasing scalar* terhadap waktu. Untuk *learning rate* menggunakan persamaan 3.8. Dimana, *T* adalah adalah maksimum epoch dan *t* adalah epoch pada saat itu, $\alpha(0)$ learning rate awal (LRStart), $\alpha(T)$ learning rate akhir (LRStop).

$$\alpha(t) = \alpha(0) \left(\frac{\alpha(T)}{\alpha(0)}\right)^{\frac{t}{T}} \tag{3.8}$$

Pada tahap awal jumlah node yang aktif sangat sedikit sehingga akan terjadi perubahan bobot yang signifikan disaat awal, untuk mengatasi hal tersebut metode GSOM pada saat proses update vector referensi di kalikan dengan $\psi_{(n)}$ yang merupakan suatu fungsi dari jumlah node yang terdapat pada map (Alahakoon, 2000) dan dinyatakan dengan persamaan 3.9.

$$\psi_{(n)} = 1 - \frac{R}{n(t)} \tag{3.9}$$

Dimana $n(t)$ merupakan jumlah node yang aktif dan R adalah konstanta yang bisa ditentukan dengan nilai 3,8, hal ini di ambil karena jumlah node awal adalah 4. Sehingga pada epoch pertama persamaan untuk update bobot vector referensi dari persamaan 3.6 menjadi persamaan 3.10.

$$\boldsymbol{w}_j(t+1) = \boldsymbol{w}_j(t) + \psi_{(n)}\, h_{win\ j}(\mathbf{x}(\mathrm{t}) - \boldsymbol{w}_j(t)) \tag{3.10}$$

3.5 Skenario pengujian

Pengujian pada penelitian ini menggunakan spesifikasi komputer sebagai berikut:

- Processor : Intel® Core™ i7-6500U
- CPU : 2 Core(s) @2.50 GHz, 2601 Mhz, 4 Logical Processor(s)
- Memory : 8 Gb
- Operating Sistem : Windows 64-bit

3.5.1 Pengujian RGSOM sebagai metode utama

Arsitektur pengujian penelitian pada tesis ini dapat dilihat pada Gambar 3.4. Fitur yang terdapat pada *data set* akan dilakukan tahap preprocessing yaitu pengacakan data, pemetaan kelas, label encoder, dan normalisasi. Dan kemudian dilanjutkan proses reduksi fitur dengan menggunakan metode PCA. Reduksi fitur dilakukan dengan tujuan untuk meningkatkan efisiensi pada proses klasifikasi.

Gambar 3.4 Arsitektur sistem deteksi intrusi menggunakan PCA + RGSOM

Pada pengujian menggunakan metode RGSOM akan di lakukan beberapa kali percobaan dengan metode K-Fold untuk menentukan parameter. Pada penentuan parameter data yang digunakan diambil adalah dari data latih dari *data set* KDD 1999 yang kemudian dilakukan pemisahan secara random dengan komposisi 20% sebagai data latih dan 80% sebagai data uji. Pemisahan ini menggunakan metode K-Fold dengan nilai k adalah 10, dan terdistribusi secara merata sesuai dengan komposisi pesentase tiap kelas. Hal ini ditujukan untuk menjawab rumusan permasalahan nomor 1 tentang, bagaimana parameter RGSOM untuk mencapai hasil yang optimal dalam proses deteksi terhadap intrusi.

Pada penentuan parameter, *initial map* yang digunakan sama untuk setiap percobaan agar di ketahui pengaruh perubahan parameter terhadap akurasi klasifikasi. *Initial map* yang dipergunakan menggunakan metode PCA Initialization (PCI), perhitungan bobot PCI dapat dilihat pada lampiran B (PCA INITIALIZATION). Parameter terbaik yang akan dicari antara lain adalah *learning rate* awal (*LRStart*), *learning rate* akhir (*LRStop*), *clustering threshold* (CT), radius (R), dan maksimum epoch (*ME*).

Penentuan parameter learning rate akhir (*LRStop*) menggunakan nilai *LRStart* = 0.9*, CT* = 0.5*,* R=1 *dan ME* = 2, dan range pengujian *LRStop* adalah 2.00E-02, 4.00E-04, 8.00E-06, 1.60E-07, 3.20E-09 dan kemudian dengan kelipatan 0.02. Percobaan akan dilakukan menggunakan metode K-Fold dan hasil akan dirata rata, kemudian LRStop yang menghasilkan nilai akurasi terbaik akan dipilih untuk pengujian selanjutnya.

Nilai parameter learning rate awal (*LRStart*) akan ditentukan dengan nilai *LRStart* antara 0.1 sampai 1 dengan kelipatan 0.1. Initial map dan *LRStop* yang digunakan adalah berdasarkan hasil percobaan sebelumnya sedangkan *CT* = 0.5*, R* = 1*, dan ME* = 2.

Parameter yang ditentukan kemudian adalah mencari nilai yang optimum untuk *CT* dengan nilai range antara 0.1 sampai dengan 1 dengan kelipatan 0.1. Kemudian hasilnya akan digunakan untuk menentukan nilai Radius yang akan diuji dengan range 1 sampai dengan 5. Parameter terakhir yang akan diuji adalah maksimum epoch (ME). Parameter ME akan diuji dengan range antara 1-5 dan waktu eksekusi juga akan dicatat.

Penentuan parameter optimum dilakukan dengan mempertimbangkan nilai akurasi dan waktu komputasi. Prioritas utama penentuan parameter adalah waktu komputasi dengan tidak mengabaikan hasil akurasi. Jika ada selisih akurasi dibawah 0,1 % maka akan dipilih waktu komputasi yang paling effisien walaupun nilai akurasinya lebih rendah.

Setelah mendapatkan nilai parameter yang optimal pada beberapa pengujian diatas akan dilakukan pengujian akhir dengan menggunakan parameter optimal untuk menentukan akurasi, false alarm rate, detection rate atau recall, dan precision untuk menjawab rumusan permasalahan nomor 4. Pada tahap ini juga dilakukan pencatatan waktu yang dibutuhkan pada proses pembelajaran dan pengujian untuk mengetahui waktu eksekusi dari RGSOM.

3.5.2 Pengujian SOM sebagai metode pembanding

Pada pengujian ini akan menggunakan SOM yang dimodifikasi. Modifikasi ini melakukan implementasi dari *clustering reference vector* dengan nilai parameter *clustering threshold* sama dengan nilai *growing threshold* pada percobaan RGSOM. Parameter jumlah ukuran vektor referensi atau peta topografi dan learning rate sama dengan yang digunakan pada RGSOM.

Pengukuran keakuratan dalam akurasi, *false alarm rate, detection rate* atau *recall,* dan *precision,* menggunakan metode pengukuran yang terdapat pada sub bab 2.9. Pada tahap pembelajaran dan pelabelan atau klasifikasi akan dilakukan pencatatan waktu komputasi untuk mengetahui efisiensi dari metode SOM.

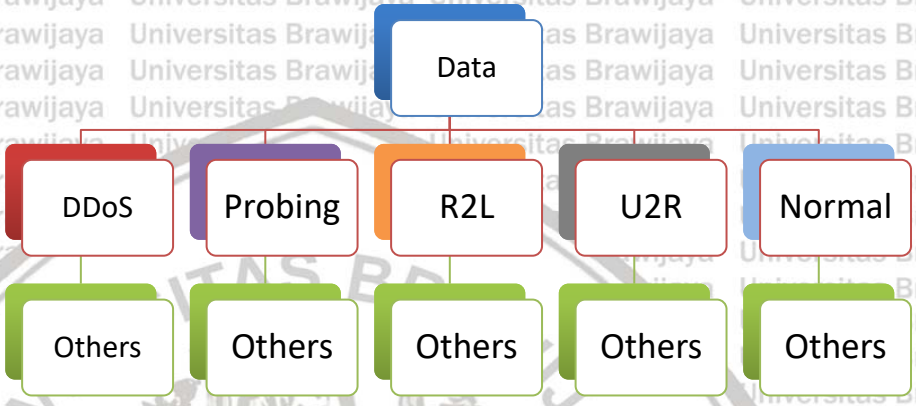
3.5.3 Pengujian GSOM sebagai metode pembanding

Metode GSOM akan dilakukan pengujian dengan menggunakan parameter *learning rate* yang sama dengan RGSOM. *Growth threshold* akan dihitung

berdasarkan persamaan 2.7 dengan *SF* sama dengan 0.5, dengan dimensi data (*D*) sesuai dengan jumlah fitur yang digunakan.

Pada proses pembelajaran dan pelabelan dilakukan pencatatan terhadap waktu komputasi untuk mengetahui efisiensi dari metode GSOM. Pengukuran akurasi, *false alarm rate, detection rate* atau *recall,* dan *precision,* menggunakan metode pengukuran yang terdapat pada sub bab 2.9.

3.5.4 Pengujian SVM sebagai metode pembanding



Gambar 3.5 Arsitektur One Versus Rest pada SVM

Dalam penelitian ini dilakukan eksperimen menggunakan metode SVM dengan kernel berbasis linier dengan pendekatan *One-Versus Rest*. Arsitektur metode SVM *One versus Rest* yang digunakan berdasarkan Gambar 3.5. Terdapat lima tahapan penggunaan SVM, dimana tahap pertama akan melakukan training dengan kelas DDoS dan selain DDoS, begitu seterusnya sampai pada tahap terakhir yaitu mengklasifikasikan label normal dan yang attack.

Tiap tahap SVM akan dicatat nilai klasifikasi yang tergolong True Positif, True Negatif, False Positif dan False Negatif. Nilai TP, TN, FP, dan FN akan digunakan untuk menentukan akurasi, false alram rate, detection rate atau recall, dan precision sesuai dengan metode yang dijelaskan pada bab 2. Waktu yang dipergunakan pada proses training dan testing akan dicatat untuk mengetahui kecepatan atau efisiensi dari SVM.

3.6 Analisis hasil dan pembahasan

Pada bagian ini akan disajikan hasil dari pengujian metode pembanding dan metode yang diajukan. Hasil pengujian akan dibandingkan dan dilakukan pembahasan berdasarkan metode pengukuran yang telah dijelaskan pada bab 2. Hasil pengujian dapat di sajikan dengan tabel dan gambar untuk mempermudah penjelasan.

BAB 4 HASIL

Berdasarkan Gambar 3.4 tentang arsitektur system deteksi intrusi jaringan dengan menggunakan metode RGSOM, semua pengujian metode menggunakan *data set*, fitur, maupun metode normalisasi yang sama. Jadi semua metode yang akan digunakan baik itu metode utama ataupun metode pembanding menggunakan data latih dan data uji yang telah di dilakakukan *preprocesing* terlebih dahulu.

4.1 Hasil reduksi fitur dengan menggunakan PCA

Data set KDD 1999 memiliki 41 fitur dan direduksi menggunakan metode PCA dengan persentase variasi yang di harapkan sebesar 95% dari data latih. Dari hasil pengujian diperoleh *principle component* sejumlah 5 (Lampiran B). Nilai dari jumlah *principle component* digunakan untuk menentukan jumlah fitur yang akan dibentuk, sehingga fitur yang semula 41 akan ditransformasikan menjadi 5 fitur (Lampiran B). Tranformasi ini di lakukan untuk fitur data training, data testing. Hasil dari PCA juga digunakan untuk membentuk *initial map* dengan menggunakan metode PCA *initialization* (PCI) (Akinduko, Mirkes, & Gorban, 2016) (Lampiran C). Hasil dari transformasi ini kemudian di proses menggunakan metode utama maupun metode pembanding.

4.2 Hasil penentuan nilai parameter pada metode RGSOM

Pada RGSOM, inisialisasi vector referensi pada peta topografi (*initial map*) menggunakan metode PCI (Principal Component Initialization) yang didapat dari kombinasi Principal Component ke 1 dan Principal Component ke 2 dari proses Principal Component Analysis. Kelebihan dari PCI adalah inisialisasi yang dihasilkan lebih terorganisasi sehingga mampu mempercepat proses komputasi jika dibandingkan dengan inisialisasi secara random (Akinduko et al., 2016).

4.2.1 Penentuan parameter learning rate akhir (LRStop)

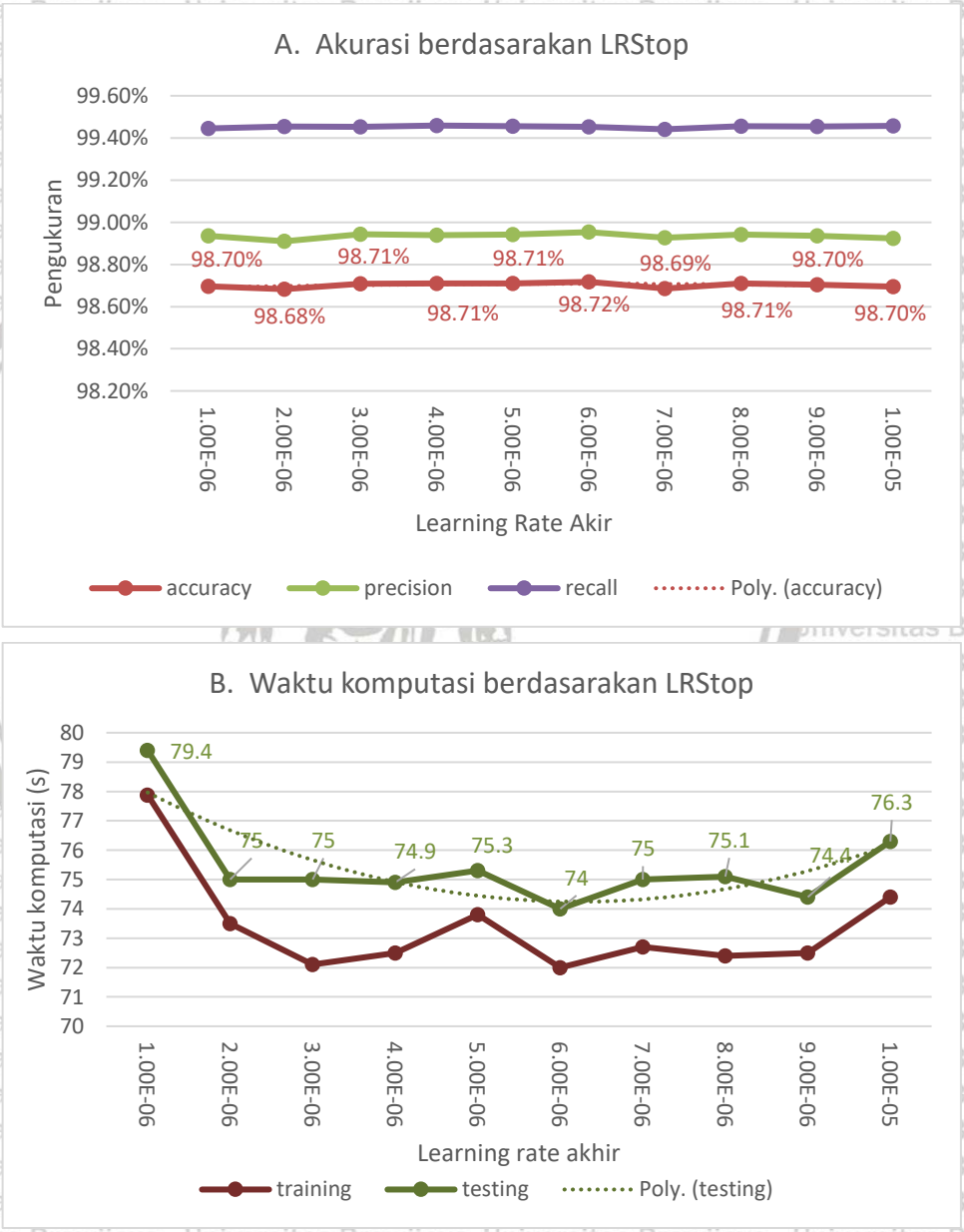
Penentuan parameter tahap pertama pada metode RGSOM adalah menentukan nilai learning rate akhir. Parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 0.9
2. Learning rate akhir = 2.00E-02, 4.00E-04, 8.00E-06, 1.60E-07, 3.20E-09
3. Radius = 1
4. Maximum Epoch = 2
5. Clustering Treshold = 0.5

Hasil pengujian penentuan learning rate akhir dapat dilihat pada Tabel 4.1 dan Gambar 4.1. Didapatkan nilai LRStop terbaik adalah 6.00E-6 dengan akurasi sebesar 98.72% dan waktu kumputasi 74 detik pada saat pengujian. Dari data tersebut perubahan learning rate akhir menghasilkan akurasi yang stabil di range learning rate akhir 1.00E-6 sampai dengan 1.00E-5.

Tabel 4.1 Penentuan parameter LRStop

stopLearningRate	accuracy	precision	recall	far	Training (s)	Testing (s)
2.00E-02	98.18%	98.37%	99.38%	6.65%	68	70.7
4.00E-04	98.55%	98.78%	99.42%	4.98%	68.3	70.4
8.00E-06	**98.71%**	98.94%	**99.46%**	4.31%	68.6	**70.2**
1.60E-07	98.66%	98.89%	99.45%	4.52%	68.5	70.5
3.20E-09	98.66%	**98.95%**	99.39%	**4.29%**	**68**	71.4



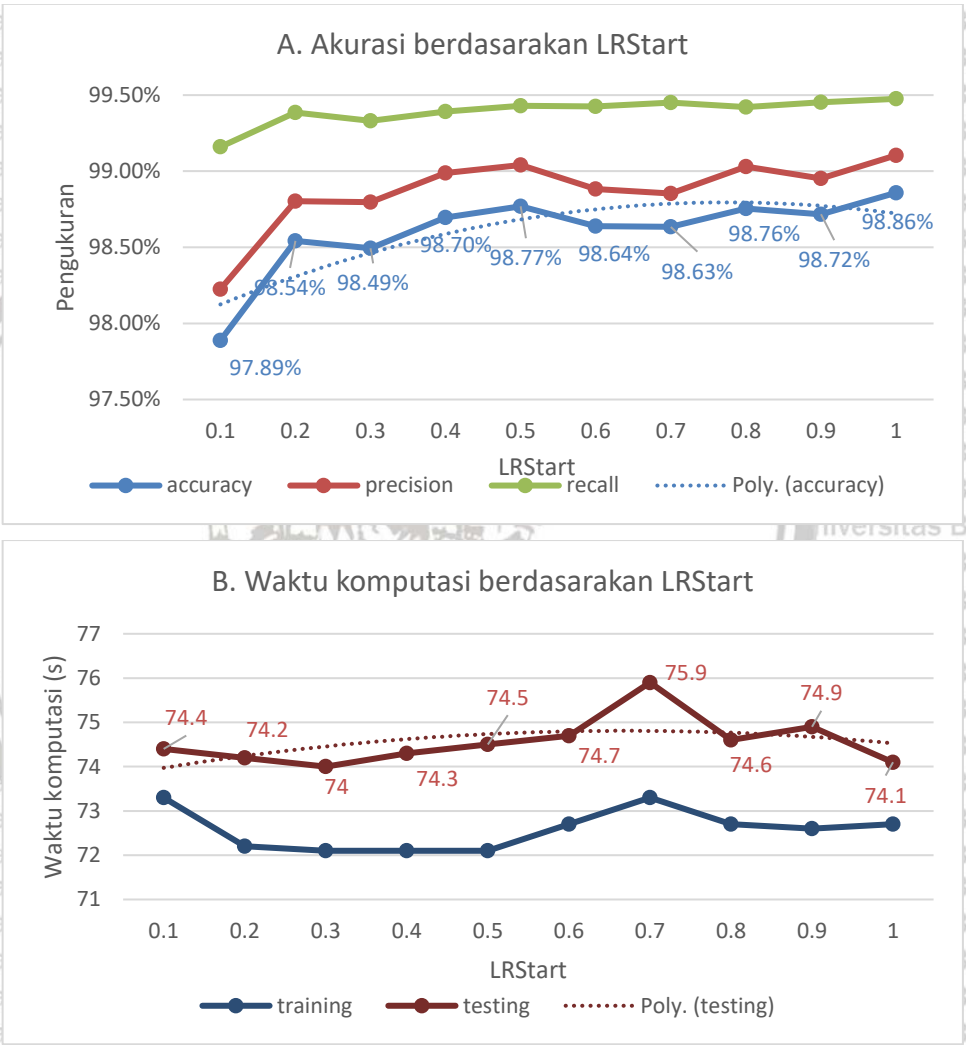
Gambar 4.1 Hasil pengujian LRStop.

A) Akurasi berdasarkan LRStop, dan B) Waktu komputasi berdasarkan LRStop.

4.2.2 Penentuan parameter learning rate awal (LRStart)

Pengujian berikutnya adalah penentuan lerning rate awal (LRStart). Pada proses pengujian ini parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 0.1 sampai dengan 1 dengan interval 0.1
2. Learning rate akhir = 6.00E-6
3. Radius = 1
4. Maximum Epoch = 2
5. Clustering Treshold = 0.5



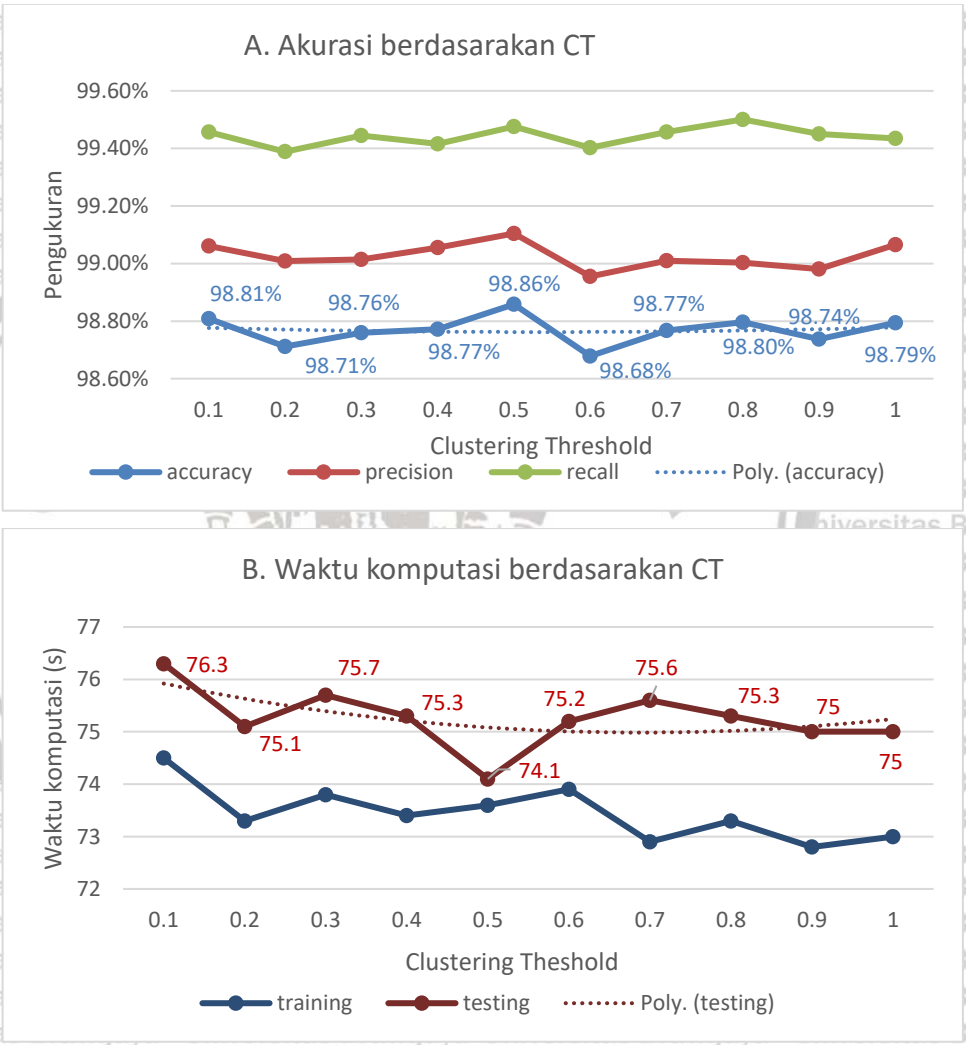
Gambar 4.2 Pengujian LRStart.
A) Akurasi berdasarakan LRStart dan B) Waktu komputasi berdasarakan LRStart

Hasil pengujian penentuan learning rate awal dapat dilihat pada Gambar 4.2. Didapatkan nilai LRStart terbaik adalah 1 dengan akurasi sebesar 98,86% dengan waktu komputasi 74.1 detik. Dari data diatas perubahan learning rate awal menghasilkan akurasi yang memiliki kecederungan meningkat dan stabil setelah LRStart mulai dari 0.2, dengan hasil percobaan memiliki nilai akurasi diatas 98%.

4.2.3 Penentuan parameter clustering threshold (CT)

Pengujian berikutnya adalah penentuan clustering threshold (CT). Pada proses pengujian ini parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 1
2. Learning rate akhir = 6.00E-6
3. Radius = 1
4. Maximum Epoch = 2
5. Clustering Treshold = 0.1 sampai dengan 1



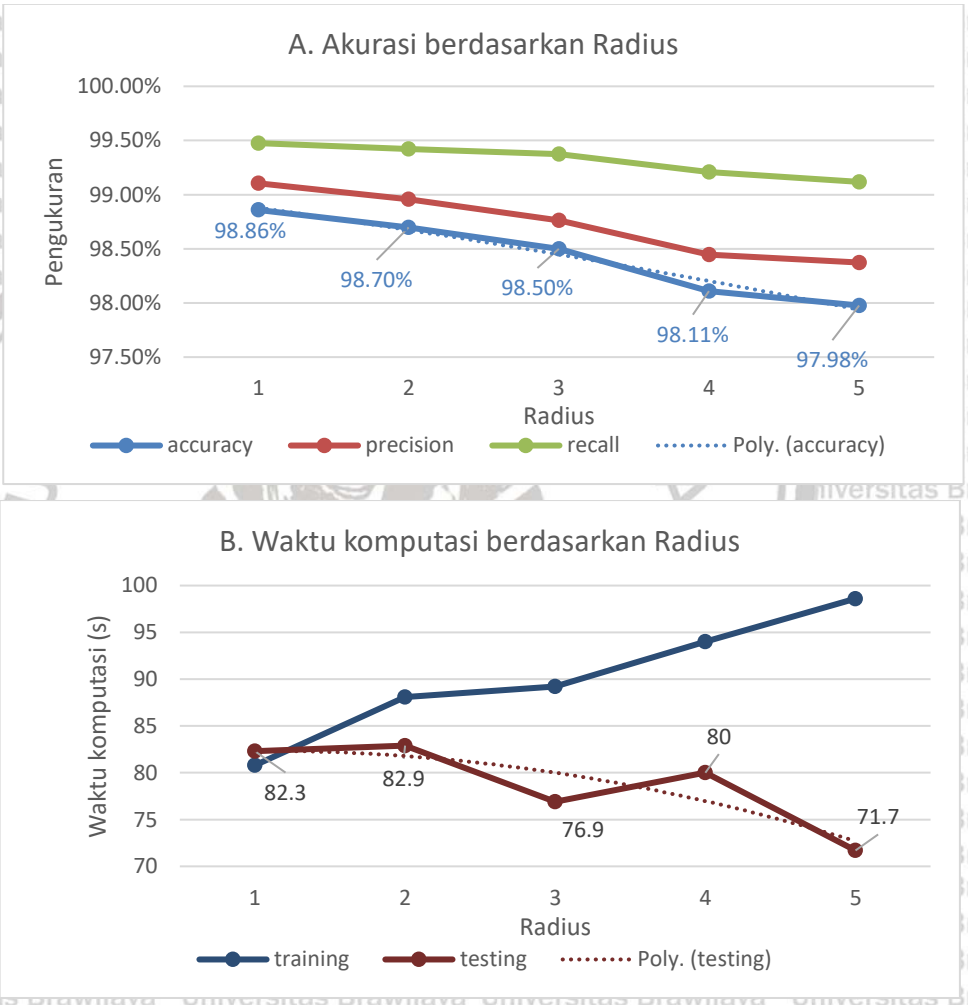
Gambar 4.3 Penentuan parameter CT.
A) Akurasi berdasarakan CT dan B) Waktu komputasi berdasarakan CT.

Hasil pengujian penentuan *clustering threshold* dapat dilihat pada Gambar 4.3. Didapatkan nilai CT terbaik adalah 0.5 dengan akurasi sebesar 98,86% dan waktu komputasi yang dibutuhkan untuk pengujian adalah 74.1 detik. Dari data diatas perubahan CT menghasilkan akurasi dengan kecenderungan stabil dengan nilai akurasi diatas 98%.

4.2.4 Penentuan parameter neighborhood radius (R)

Pengujian berikutnya adalah penentuan *neighborhood radius* (R). Pada proses pengujian ini parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 1
2. Learning rate akhir = 6.00E-6
3. Radius = 1 sampai 5 dengan interval 1
4. Maximum Epoch = 2
5. Clustering Treshold = 0.5



Gambar 4.4 Penentuan parameter Radius.

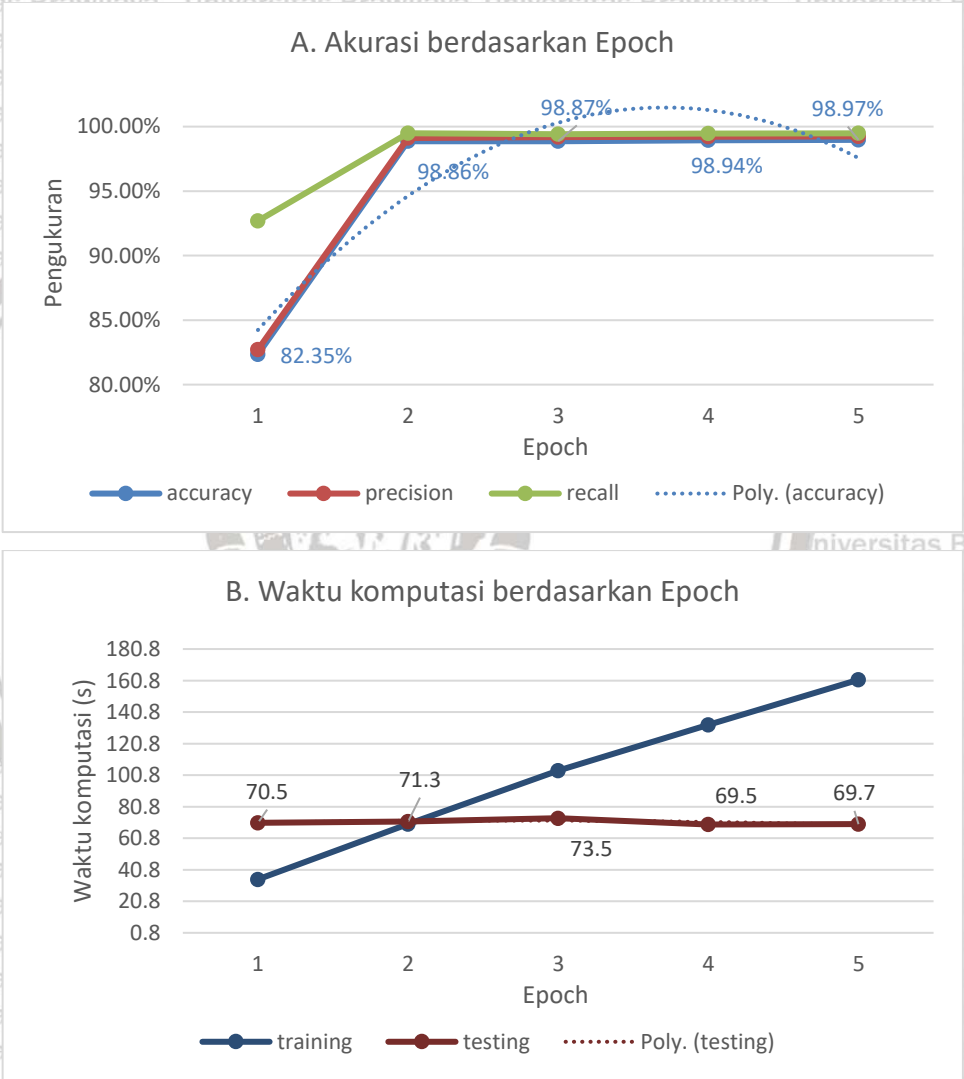
A) Akurasi dan B) Waktu komputasi berdasarkan Radius

Hasil pengujian penentuan *neighborhood radius* dapat dilihat pada Gambar 4.4. kecenderungan dari nilai akurasi dan waktu komputasi menurun berdasarkan kenaikan nilai radius. Diperoleh nilai Radius terbaik adalah 1 dengan akurasi sebesar 98.86% dan waktu komputasi yang dibutuhkan sebesar 82.3 detik.

4.2.5 Penentuan parameter maksimum epoch (ME)

Pengujian berikutnya adalah penentuan *maksimum epoch* (ME). Pada proses pengujian ini parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 1
2. Learning rate akhir = 6.00E-6
3. Radius = 1
4. Maximum Epoch = 1 sampai 5 dengan interval 1
5. Clustering Treshold = 0.5

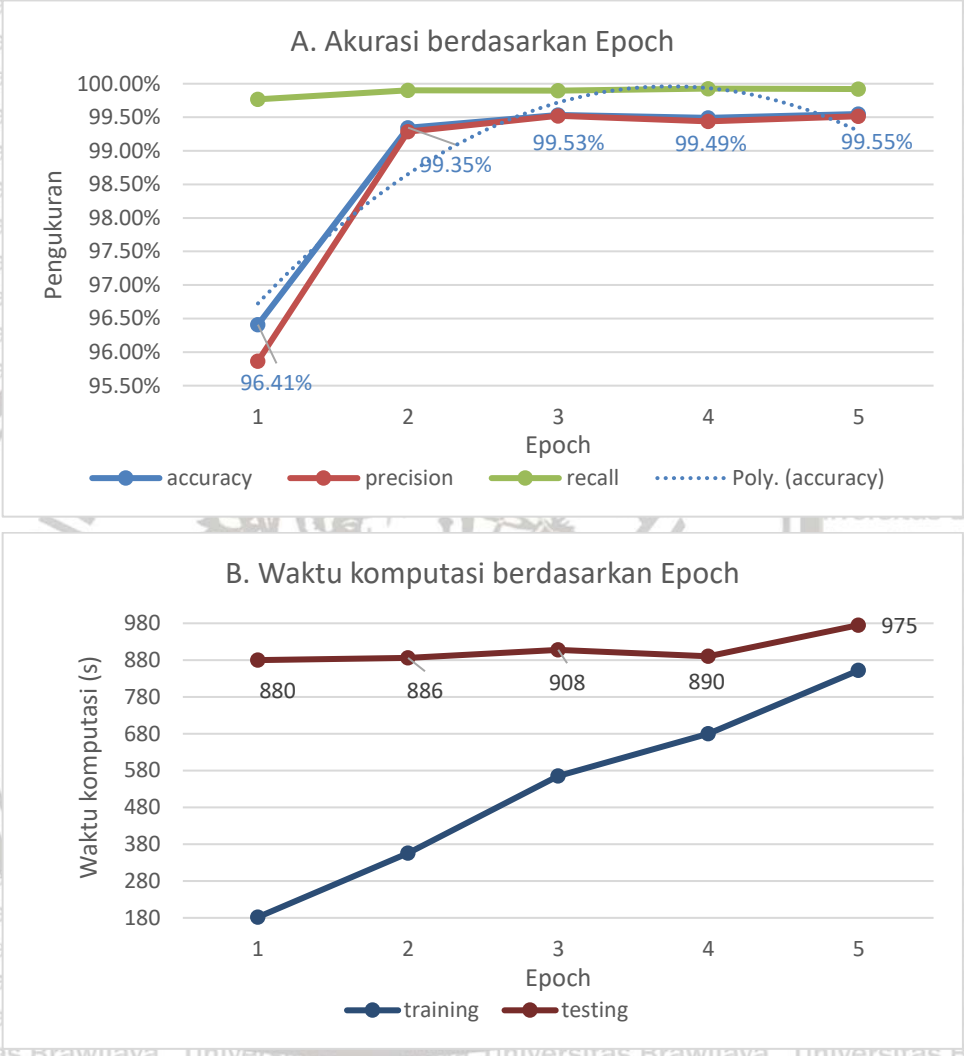


Gambar 4.5 Penentuan parameter ME dengan metode K-Fold.

A) Akurasi dan B) Waktu komputasi berdasarkan Maksimum Epoch.

Hasil pengujian penentuan parameter max epoch dapat dilihat pada Gambar 4.5, didapatkan kesimpulan bahwa perubahan Epoch menghasilkan akurasi yang memiliki kecenderungan meningkat seiring bertambahnya epoch, tetapi waktu

komputasi yang diperlukan untuk training data semakin meningkat berbanding lurus dengan kenaikan epoch, dan waktu komputasi pada saat pengujian memiliki kecenderungan stabil. Pada tahap penentuan parameter ME, ada dua nilai yang memungkinan untuk diambil yaitu epoch 2 dan 3. Untuk memastikan nilai mana yang akan diambil, kemudian dilakukan pengujian parameter dengan menggunakan 100% data latih dan 100% data uji yang terdapat pada *data set* KDD 1999.



Gambar 4.6 Penentuan parameter ME dengan 100% data latih dan 100% data uji pada *data set* KDD 1999.

A) Akurasi dan B) Waktu komputasi berdasarkan Maksimum Epoch.

Berdasarkan data yang diperoleh pada Gambar 4.6 didapatkan nilai ME yang optimum adalah 3 dengan akurasi sebesar 99,53%, dengan waktu komputasi untuk pengujian sebesar 908 detik.

4.3 Hasil pengujian dengan menggunakan metode RGSOM

Pengujian berikutnya adalah pengujian RGSOM dengan parameter yang dihasilkan dari penentuan parameter menggunakan K-Fold. Pada proses pengujian ini parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 1
2. Learning rate akhir = 6.00E-6
3. Radius = 1
4. Maximum Epoch = 3
5. Clustering Treshold = 0.5

Dari pengujian dengan menggunakan metode RGSOM di peroleh waktu komputasi rata-rata yang diperlukan untuk melakukan pelatihan (training) adalah 1219.8 detik, dan waktu komputasi yang digunakan untuk melakukan klasifikasi (testing) adalah 3031.2 detik. Berdasarkan Tabel 4.13 dapat diketahui bahwa akurasi dari metode RGSOM sebesar 99.70 %, dengan rasio deteksi (DTR) atau recal sebesar 99.70%, dan precision sebesar 99.93%, dan false alarm rate sebesar 1.22%.

Tabel 4.2 Pengukuran akurasi hasil pengujian dengan metode RGSOM

Exp	accuracy	precision	recall	far	Training (s)	Testing (s)
1	99.70%	99.93%	99.70%	1.22%	1238	3114
2	99.70%	99.93%	99.70%	1.22%	1176	2976
3	99.70%	99.93%	99.70%	1.22%	1239	2984
4	99.70%	99.93%	99.70%	1.22%	1266	3177
5	99.70%	99.93%	99.70%	1.22%	1180	2905
AVG	**99.70%**	**99.93%**	**99.70%**	**1.22%**	**1219.8**	**3031.2**

Tabel 4.3 Confusion Matrix hasil pengujian dengan metode RGSOM

Predicted	dos	normal	probe	r2l	u2r	__all__
Actual						
dos	3882105	1139	9	114	3	3883370
normal	6107	961346	1716	3343	269	972781
probe	44	1356	39694	8	0	41102
r2l	18	102	7	798	201	1126
u2r	0	14	2	9	27	52
__all__	3888274	963957	41428	4272	500	4898431

Dari Tabel 4.3 yang menunjukkan tentang *confusion matrix* dari pengujian metode RGSOM tanpa menggunakan PCA sebagai reduksi fitur dan mampu mendeteksi jenis serangan u2r sebanyak 27 dari 52 kejadian.

4.4 Hasil pengujian dengan menggunakan metode PCA + RGSOM

Pengujian berikutnya adalah pengujian PCA + RGSOM dengan parameter yang telah didapat dari pengujian sebelumnya. Pada proses pengujian ini parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 1
2. Learning rate akhir = 6.00E-6
3. Radius = 1
4. Maximum Epoch = 3
5. Clustering Treshold = 0.5

Dari pengujian dengan menggunakan metode PCA + RGSOM di peroleh waktu komputasi rata-rata yang diperlukan untuk melakukan pelatihan (training) adalah 519.4 detik, dan waktu komputasi yang digunakan untuk melakukan klasifikasi (testing) adalah 884.8 detik. Berdasarkan Tabel 4.4 dapat diketahui bahwa akurasi dari metode PCA + RGSOM sebesar 99.53 %, dengan rasio deteksi (DTR) atau recal sebesar 99.90%, dan precision sebesar 99.90%, dan false alarm rate sebesar 1.93%.

Tabel 4.4 Pengukuran akurasi hasil pengujian dengan metode PCA + RGSOM

Exp	accuracy	precision	recall	far	Training (s)	Testing (s)
1	99.53%	99.52%	99.90%	1.93%	527	889
2	99.53%	99.52%	99.90%	1.93%	529	874
3	99.53%	99.52%	99.90%	1.93%	511	885
4	99.53%	99.52%	99.90%	1.93%	510	890
5	99.53%	99.52%	99.90%	1.93%	520	886
AVG	**99.53%**	**99.52%**	**99.90%**	**1.93%**	**519.4**	**884.8**

Tabel 4.5 Confusion Matrix hasil pengujian dengan metode PCA + RGSOM

Predicted	dos	normal	probe	r2l	u2r	__all__
Actual						
dos	3881110	2015	151	42	52	3883370
normal	8785	955382	1554	5473	1587	972781
probe	1097	1712	38285	1	7	41102
r2l	7	278	1	793	47	1126
u2r	2	29	0	14	7	52
__all__	3891001	959416	39991	6323	1700	4898431

Dari Tabel 4.5 yang menunjukkan tentang *confusion matrix* dari pengujian metode PCA + RGSOM yang menggunakan PCA untuk reduksi fitur mampu menghasilkan akurasi yang tinggi (99.53%) dan metode ini mampu mendeteksi jenis serangan u2r sebanyak 7 dari 52 kejadian.

4.5 Hasil pengujian dengan menggunakan metode SVM

Berikut ini adalah hasil pengujian menggunakan metode SVM dengan menggunakan 41 fitur tanpa dilakukan reduksi fitur. Parameter yang digunakan pada pengujian menggunakan metode SVM adalah sebagai berikut:

a. Kernel : RBF
b. Mode klasifikasi : One Versus Rest
c. SVM regularization : 0.1
d. Gamma : Auto

Dari pengujian dengan menggunakan metode SVM di peroleh waktu komputasi yang diperlukan untuk melakukan pelatihan (training) adalah 4514 detik, dan waktu komputasi yang digunakan untuk melakukan klasifikasi (testing) adalah 9213 detik. Berdasarkan Tabel 4.6 dapat diketahui bahwa akurasi dari metode SVM sebesar 99.83 %, dengan rasio deteksi (DTR) sebesar 99.84%, dan precision sebesar 99.94%, dan false alarm rate sebesar 0.25%.

Tabel 4.6 Pengukuran akurasi hasil pengujian dengan metode SVM

ACC	DTR	Precision	FAR	Training (s)	Testing (s)
99.83%	99.84%	99.94%	0.25%	4514	9213

Tabel 4.7 Confusion matrix hasil pengujian dengan metode SVM

Predicted	dos	normal	probe	r2l	u2r	__all__
Actual						
dos	3880544	2799	22	5	0	3883370
normal	127	970586	113	1955	0	972781
probe	157	3140	37805	0	0	41102
r2l	10	174	1	941	0	1126
u2r	1	36	0	15	0	52
__all__	3988615	909812	4	0	0	4898431

Dari Tabel 4.7 metode SVM yang menggunakan 41 fitur tanpa dilakukan reduksi mampu menghasilkan akurasi yang tinggi (99.83%) tetapi metode ini gagal dalam mendeteksi jenis serangan u2r.

4.6 Hasil pengujian dengan menggunakan metode PCA + SVM

Berikut ini adalah hasil pengujian menggunakan metode SVM yang fitur yang terdapat pada *data set* direduksi terlebih dahulu dengan menggunakan PCA. Parameter yang digunakan pada pengujian menggunakan metode PCA + SVM adalah sebagai berikut:

a. Kernel : RBF
b. Mode klasifikasi : One Versus Rest
c. SVM regularization : 0.1
d. Gamma : Auto

Dari pengujian dengan menggunakan metode PCA + SVM di peroleh waktu komputasi yang diperlukan untuk melakukan pelatihan (training) adalah 3692 detik, dan waktu komputasi yang digunakan untuk melakukan klasifikasi (testing) adalah 7030 detik. Berdasarkan Tabel 4.8 dapat diketahui bahwa akurasi dari metode SVM sebesar 99.83 %, dengan rasio deteksi (DTR) sebesar 99.84%, dan precision sebesar 99.94%, dan false alarm rate sebesar 0.25%.

Tabel 4.8 Pengukuran akurasi hasil pengujian dengan metode PCA + SVM

ACC	DTR	Precision	FAR	Training (s)	Testing (s)
99.83%	99.84%	99.95%	0.22%	3692	7030

Tabel 4.9 Confusion matrix hasil pengujian dengan metode PCA + SVM

Predicted	dos	normal	probe	r2l	u2r	__all__
Actual						
dos	3879365	3811	191	3	0	3883370
normal	413	971363	115	890	0	972781
probe	500	1811	38791	0	0	41102
r2l	12	483	0	631	0	1126
u2r	0	34	0	18	0	52
__all__	3880290	977502	39097	1542	0	4898431

Dari Tabel 4.9 metode PCA + SVM yang menggunakan reduksi fitur mampu menghasilkan akurasi yang tinggi (99.83%) tetapi metode ini gagal dalam mendeteksi jenis serangan u2r.

4.7 Hasil pengujian dengan menggunakan metode PCA + SOM

Pengujian berikutnya adalah dengan menggunakan metode SOM dengan penggunaan *clustering reference vector* (CRV). Pada proses pengujian menggunakan metode som dengan CRV, parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 0.9
2. Learning rate akhir = 0.08
3. Radius = 1
4. Maximum Epoch = 4
5. Ukuran map = 10x10 sampai dengan 50x50
6. Clustering Treshold = 0.5

Berdasarkan Tabel 4.10 waktu komputasi yang dipergunakan pada metode PCA + SOM dengan CRV meningkat seiring dengan bertambahnya ukuran peta topografi, dan akurasi yang dihasilkan cenderung meningkat. Hasil terbaik diperoleh dengan menggunakan peta berukuran 50x50 dengan akurasi sebesar 98.00%, tetapi waktu komputasi yang diperlukan sebesar 24297 detik untuk pelatihan dan untuk testing waktu yangdiperlukan sebesar 63071 detik.

Tabel 4.10 Pengukuran akurasi hasil pengujian dengan metode PCA + SOM

Size	ACC	DTR	Precision	FAR	Training	Testing
10x10	40.27%	32.31%	81.90%	28.26%	1480	3128
20x20	88.54%	87.52%	97.95%	7.36%	4363	11808
30x30	46.88%	35.07%	96.17%	5.61%	11734	29018
40x40	95.32%	96.79%	97.28%	10.24%	19187	40609
50x50	98.00%	99.50%	98.03%	8.07%	24297	63071

Tabel 4.11 Confusion matrix hasil pengujian dengan metode PCA + SOM menggunakan peta 50x50

Predicted	dos	normal	probe	r2l	u2r	__all__
Actual						
dos	3871198	11812	67	293	0	3883370
normal	64867	895186	2713	7547	2468	972781
probe	366	7002	33728	2	4	41102
r2l	49	658	23	202	194	1126
u2r	4	32	0	14	2	52
__all__	3936484	914690	36531	8058	2668	4898431

Dari Tabel 4.11 memperlihatkan *confusion matrix* dari hasil pengujian metode PCA + SOM dengan CRV dengan ukuran peta 50x50. Metode ini mampu melakukan deteksi terhadap u2r walaupun jumlahnya sedikit (2).

4.8 Hasil pengujian dengan menggunakan metode PCA + GSOM

Pengujian berikutnya adalah dengan menggunakan metode PCA + GSOM. Pada proses pengujian menggunakan metode GSOM, parameter yang dipergunakan adalah sebagai berikut:

1. Learning rate awal = 0.9
2. Alpha = 0.2
3. Maximum Epoch = 4
4. Spread factor = 0.5

Tabel 4.12 Pengukuran akurasi hasil pengujian dengan metode PCA + GSOM

Exp	ACC	DTR	Precision	FAR	Training	Testing
1	99.58%	99.61%	99.86%	1.56%	5201	9158
2	**99.64%**	99.67%	99.88%	1.33%	**5108**	9177
3	99.62%	99.66%	99.86%	1.37%	5393	9184
4	99.62%	**99.68%**	99.84%	**1.29%**	5235	**9123**
5	99.49%	99.47%	**99.89%**	2.12%	5351	9160
Averange	99.59%	99.62%	99.87%	1.53%	5257.6	9160.4

Tabel 4.13 Confusion matrix hasil pengujian dengan metode PCA + GSOM pada percobaan ke 2.

Predicted	dos	normal	probe	r2l	u2r	__all__
Actual						
dos	3880785	2267	318	0	0	3883370
normal	5573	962137	737	4334	0	972781
probe	2022	1343	37722	15	0	41102
r2l	12	1090	2	22	0	1126
u2r	0	52	0	0	0	52
__all__	3888392	966889	38779	4371	0	4898431

Berdasarkan Tabel 4.12 waktu komputasi yang dipergunakan pada metode PCA + GSOM rata rata adalah 5257.6 detik untuk pelatihan dan 9160.4 detik untuk testing. Akurasi rata rata dari 5 kali percobaan adalah 99,59%, dengan detection rate 99.62%, dan precision sebesar 99.87% dan flase alarm rate adalah 1.53%. Dari Tabel 4.13 memperlihatkan *confusion matrix* dari hasil pengujian metode PCA + GSOM dan metode ini tidak mampu melakukan deteksi terhadap u2r.

BAB 5 PEMBAHASAN

Dari hasil pengujian pada bab sebelumnya akan dilakukan pembahasan terkait hasil dari masing masing metode dan perbandingan antar metode meliputi waktu komputasi dan akurasi pengukuran.

5.1 Nilai parameter terbaik metode RGSOM

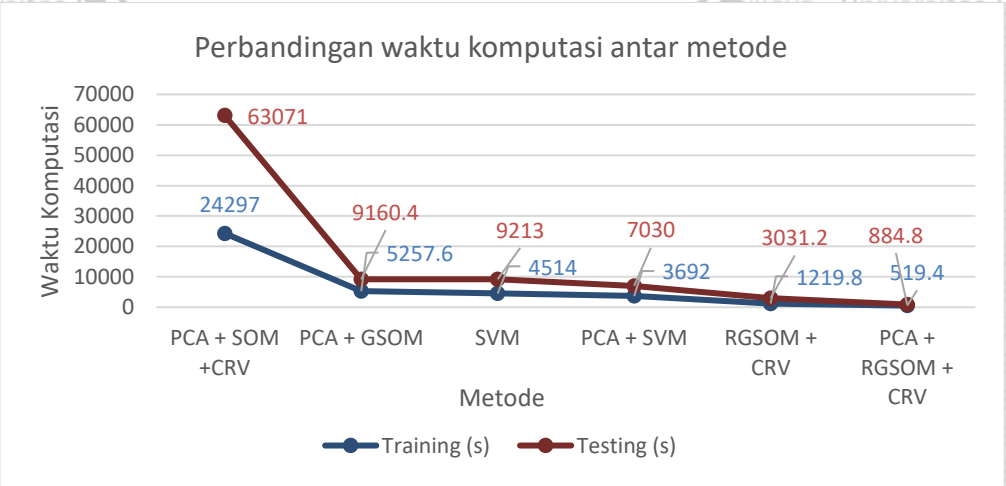
Penentuan nilai parameter terbaik dilakukan dengan menggunakan metode K-Fold dengan nilai K adalah 10. Data yang dipergunakan adalah data latih pada *data set* KDD 1999 sebanyak 494.021 yang di bagi dengan komposisi 20% akan digunakan sebagai data latih dan 80% sebagai data uji pada proses penentuan parameter terbaik. Metode K-Fold melakukan pembagian tersebut dan menentukan instance secara acak sebanyak nilai K-nya. Penggunaan metode ini mampu menunjukkan kecenderungan nilai parameter yang terbaik dalam tiap tahap penentuan parameter, hal ini disebabkan data yang ambil secara acak dan memiliki komposisi yang berbeda beda dan hasilnya di rata-rata.

Berdasarkan hasil pengujian penentuan parameter terbaik untuk menghasilkan akurasi yang optimal pada metode PCA + RGSOM yang di lakukan dengan metode K-Fold didapatkan nilai parameter sebagai berikut:

1. Learning rate awal = 1
2. Learning rate akhir = 6.00E-6
3. Radius = 1
4. Maximum Epoch = 3
5. Clustering Treshold = 0.5

Dengan parameter tersebut didapatkan hasil akurasi sebesar 99,53% dengan waktu komputasi rata-rata 519,4 detik untuk pelatihan dan 884,8 detik untuk pengujian.

5.2 Perbandingan efisiensi waktu komputasi

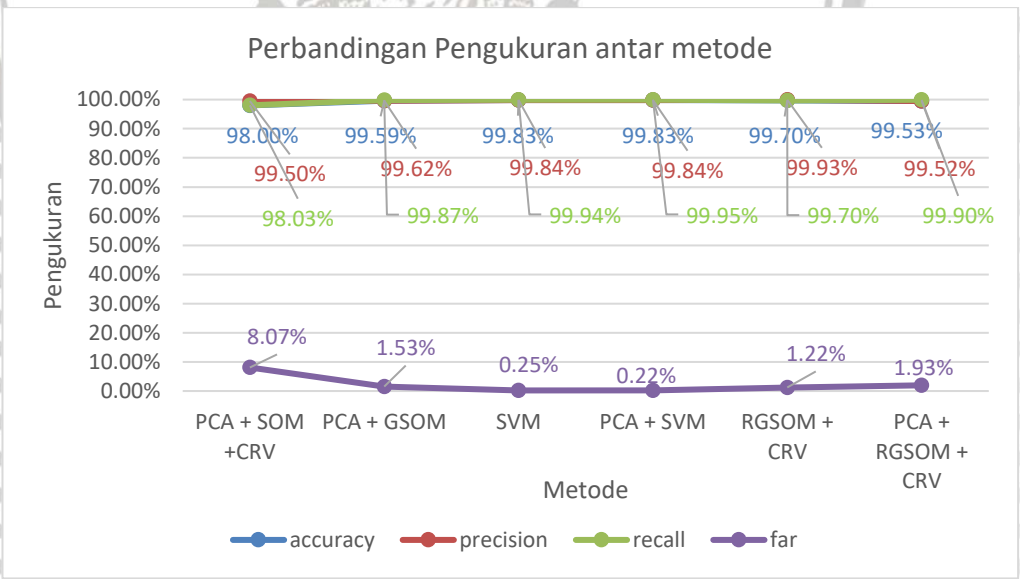


Gambar 5.1 Perbandingan waktu komputasi antar metode

Berdasarkan Gambar 5.1 yang menunjukkan grafik waktu komputasi pada saat pelatihan dan pengujian masing masing metode, mengkonfirmasi latar belakang penulisan tesis ini tentang lamanya waktu komputasi metode SOM yang berusaha di perbaiki dengan metode GSOM. GSOM memiliki kecepatan 4,6x lebih cepat dari pada metode SOM pada saat pelatihan, dan 6,8x lebih cepat pada saat pengujian. Metode utama yang diusulkan (PCA + RGSOM) mampu memperbaiki waktu komputasi dari metode GSOM, menjadi 10,1x lebih cepat pada saat pelatihan dan 10,3x lebih cepat pada saat pengujian. Jika dibandingkan dengan SOM, maka waktu komputasi PCA + RGSOM lebih cepat 46,7x pada saat pelatihan dan 71,2x pada saat pengujian. Waktu komputasi metode utama yang diusulkan (PCA + RGSOM) juga lebih cepat 7,1x dari pada metode SVM pada saat pelatihan dan 7,9x pada saat pengujian yang sama sama melalui proses reduksi fitur.

Kecepatan waktu komputasi pada metode RGSOM dikarenakan *node* pada peta topografi berkembang secara adaptif berdasarkan data yang masuk pada saat pembelajaran. Pada saat pengujian kecepatan juga meningkat secara signifikan karena jumlah node yang di bandingkan hanya terbatas pada *node* yang memiliki kelas (*used node*). Pada proses penghitungan jarak *eucledian* pada penelitian ini juga menggunakan manipulasi array dengan numpy pada bahasa pemrograman python. Penggunaan metode RGSOM ditunjang dengan kemampuan manipulasi array numpy menghasilkan kecepatan waktu komputasi pada penelitian ini paling efisien jika dibandingkan dengan metode pembanding yang lainnya.

5.3 Perbandingan akurasi



Gambar 5.2 Perbandingan akurasi, *precision*, *recall* dan FAR antar metode.

Berdasarkan Gambar 5.2 yang menunjukkan grafik perbandingan pengukuran akurasi antar metode. Metode pengukuran akurasi sesuai dengan yang terdapat pada bab 2.9, yang memiliki 4 macam pengukuran yaitu akurasi, *detection rate*, *precision*, dan *false alrm rate*. Dalam penelitian ini tidak menggunakan akurasi saja sebagai metode pengukuran, hal ini dikarenakan data yang terdapat dalam *data*

set KDD 1999 merupakan data dengan tipe *unbalance*. Dimana data dengan tipe ini memiliki distribusi yang tidak merata pada tiap kelasnya. Jika yang di pergunakan hanya akurasi maka berdasarkan data pada Tabel 4.1 jika semua data uji di beri label sebagai **dos** semua, akurasinya sudah berkisar 79%. Jadi diperlukan metode pengukuran yang lainnya.

Jika dilihat dari Gambar 5.2 semua metode memiliki akurasi diatas 90%, dan sebagian besar memiliki akurasi diatas 99%, kecuali metode PCA + SOM + CRV yang memiliki akurasi 98%. Jika dilihat dari akurasi, DTR, *precision*, dan FAR, metode PCA + SVM memiliki hasil yang terbaik diantara metode yang di ujikan. Metode PCA + SVM menghasilkan akurasi sebesar 99,88% dengan DTR sebesar 99,84%, precision sebesar 99,95% dan FAR hanya 0,22%. Metode yang diusulkan lebih rendah 0.30% dalam hal akurasi, lebih rendah 0.05% dalam DTR, lebih rendah 0.32% dalam presision, dan lebih rendah 1.71% pada FAR.

Apabila kita melihat Tabel 4.3 c*onfusion matrix* hasil pengujian dengan metode RGSOM dan dibandingkan dengan Tabel 4.7 *confusion matrix* hasil pengujian dengan metode PCA + SVM, kita ketahui bahwa metode yang di usulkan mampu melakukan deteksi terhadap serangan u2r, sedangkan metode SVM yang direduksi fiturnya tidak mampu melakukan deteksi terhadap serangan u2r, hal ini dikarenakan karakteristik data *unbalance,* dimana data yang berlabel u2r jumlahnya sangat sedikit dibandingkan dengan label yang lain. Pada Tabel 4.11 Confusion Matrix hasil pengujian dengan metode PCA + GSOM pada percobaan ke 2, metode PCA + GSOM juga tidak mampu melakukan deteksi terhadap serangan u2r. Pada metode RGSOM dan GSOM kita akan melihat visualisasi model yang terbentuk pada proses pelatihan untuk menganalisa mengapa RGSOM mampu mendeteksi serangan u2r sedangkan metode GSOM tidak mampu mendeteksi serangan tersebut.

5.4 Pengaruh reduksi fitur dengan menggunkan PCA

Berdasarkan hasil penelitian yang disajikan pada Gambar 5.1 dan Gambar 5.2 penggunaan metode RGSOM tanpa reduksi fitur (menggunakan 41 fitur) menghasilkan akurasi 99,70% dengan waktu komputasi sebesar 1219,8 detik untuk pelatihan dan 3031,2 detik untuk pengujian. Metode RGSOM dengan menggunakan reduksi fitur mampu menghasilkan nilai akurasi sebesar 99,53% dengan waktu komputasi sebesar 519,4 detik untuk pelatihan dan 884,8 detik untuk pengujian.

Dalam penelitian ini juga dilakukan pengujian terhadap metode SVM tanpa reduksi fitur dan SVM dengan PCA sebagai metode reduksi fitur. Pada metode SVM tanpa reduksi fitur, waktu komputasi lebih lama jika dibandingkan dengan metode SVM yang menggunakan reduksi fitur dengan PCA. Metode PCA + SVM waktu komputasinya 1.2x lebih cepat pada proses pelatihan dan 1,3x lebih cepat pada saat pengujian, jika dibandingkan dengan metode SVM tanpa reduksi fitur.

Dari hasil ini, reduksi fitur dengan menggunakan PCA terbukti efektif dalam menurunkan waktu komputasi dan hasil akurasi untuk metode SVM dengan atau

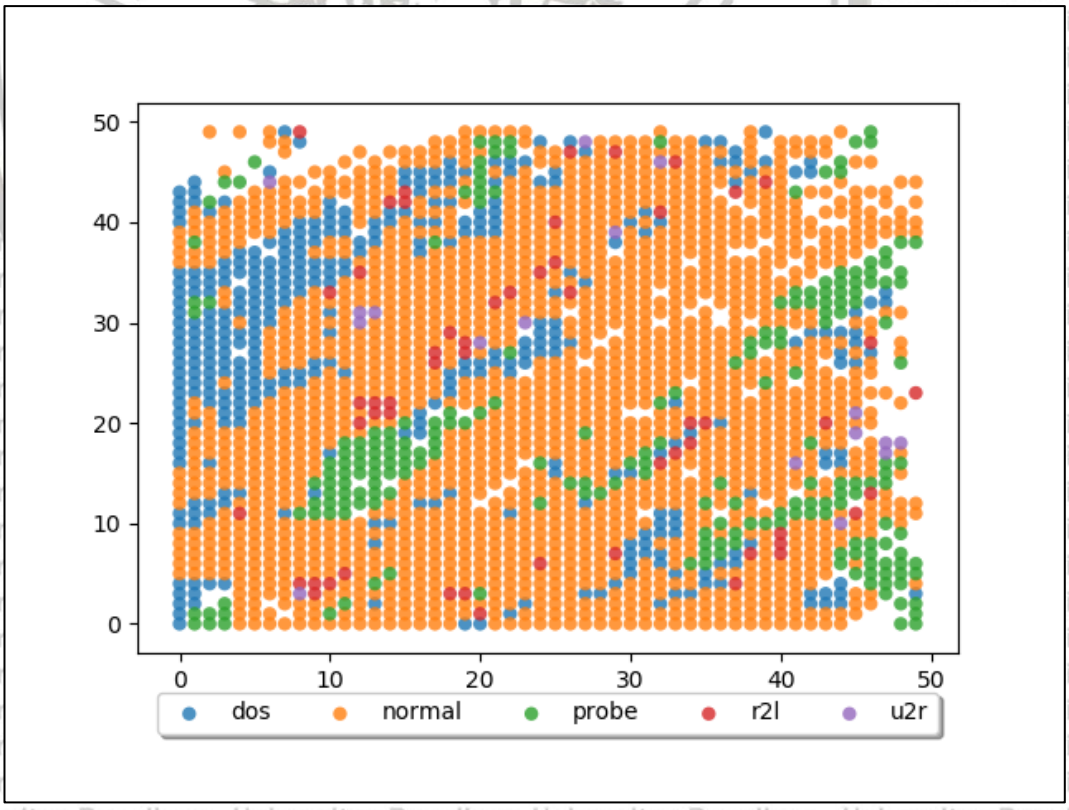
tanpa reduksi fitur tidak berbeda. Metode RGSOM yang menggunakan reduksi fitur akurasinya menurun hanya sebesar 0.17 % dibandingkan tanpa reduksi fitur, sedangkan waktu komputasi lebih cepat 3,5x pada saat pengujian dan lebih cepat 2,43x pada saat pembelajaran.

Waktu komputasi yang lebih cepat pada penerapan reduksi fitur menggunakan PCA di karenakan dengan menurunnya jumlah fitur dari 41 menjadi 5, menyebabkan waktu komputasi pada saat menghitung jarak *Euclidean* lebih cepat, hal ini juga sejalan dengan penelitian yang dilakukan oleh Ikram (2016) yang menyatakan bahwa dengan direduksinya fitur akan meminimalkan waktu pembelajaran dan pengujian.

5.5 Visualisasi peta topografi

Salah satu kelebihan dari metode SOM dan turunannya adalah metode ini mampu memvisualisasikan model yang terbentuk pada proses pelatihan baik dalam 2 dimensi maupun 3 dimensi. Dalam penelitian ini metode SOM, GSOM, dan RGSOM akan divisualisasikan dalam bentuk 2 dimensi. Visualisasi ini bertujuan untuk memberikan pemahaman lebih atas model yang dibangun pada proses pelatihan.

5.5.1 Visualisasi model PCA + SOM

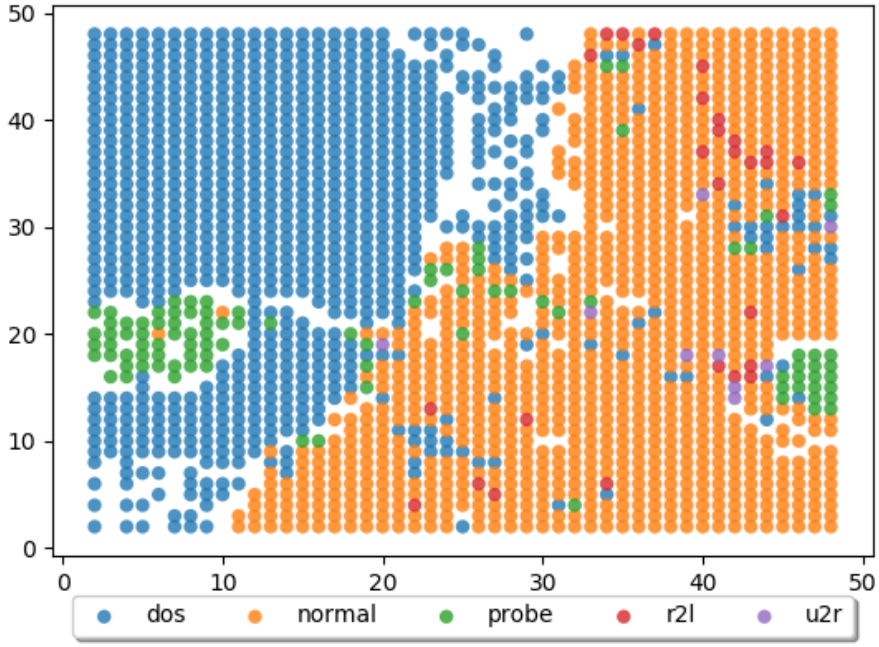


Gambar 5.3 Peta topografi PCA SOM CRV

Gambar 5.3 merupakan visualisasi model yang terbentuk dari metode SOM dengan PCA sebagai reduksi fitur dan menggunakan CRV, peta topografi sebesar

50x50 yang menghasilkan akurasi sebesar 98.00%. Apa bila kita lihat dari model yang terbentuk dari pelatihan ada beberapa node yang diberi label u2r, sehingga ini memungkinkan untuk dapat mendeteksi serangan u2r pada saat pengujian. Metode ini mampu melakukan deteksi terhadap serangan u2r walaupun jumlahnya serangan yang mampu terdeteksi adalah 2 buah dari 52 serangan.

5.5.2 Visualisasi model PCA + RGSOM



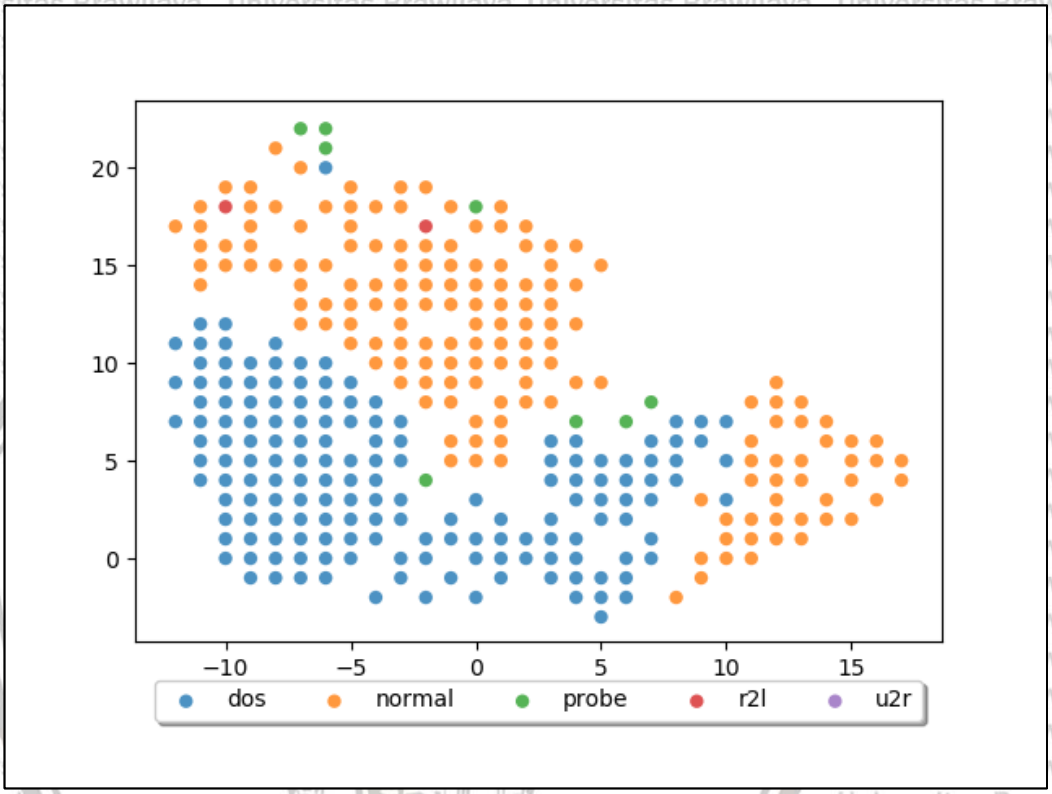
Gambar 5.4 Peta topografi PCA + RGSOM

RGSOM dengan reduksi fitur menggunakan PCA dan metode CRV untuk membuat model pada pelatihan menghasilkan visualisasi model seperti Gambar 5.4. Pada model tersebut terdapat beberapa node yang berlabel u2r, sehingga diharapkan mampu untuk mendeteksi serangan u2r. Dari Tabel 4.3 *confusion matrix* hasil pengujian dengan metode RGSOM, didapatkan bahwa metode ini mampu mendeteksi serangan u2r sebanyak 7 dari 52 serangan yang diujikan.

5.5.3 Visualisasi model PCA + GSOM

Visualisasi model yang terbentuk pada pelatihan dengan menggunakan metode GSOM dengan reduksi fitur dapat dilihat pada Gambar 5.5. Jika dilihat dari node yang terbentuk tidak terdapat node yang berlabel u2r. Dari visualisasi ini kita bisa mengetahui bahwa model dari hasil pelatihan ini tidak akan pernah bisa mengklasifikasikan serangan u2r sebagai u2r. Pada GSOM tidak di implementasi metode CRV berbeda dengan pada pengujian metode SOM dan RGSOM yang di lengkapi dengan metode CRV.

Pada metode standart dari GSOM pelabelan node pada saat pelatihan dilakukan dengan cara menguji jarak terdekat setiap node dengan setiap data latih. Jika node memiliki jarak terdekat dengan data latih dengan label **dos,** maka node tersebut akan diberi label **dos**. Jadi pada model yang divisualisasikan pada Gambar 5.5 tidak ada data latih yang berlabel u2r yang terdekat dengan node yang terbentuk pada saat pelatihan. Hal ini membuat model tidak akan pernah bisa mengklasifikasikan serangan sebagai u2r. Hal ini berbeda dengan metode CRV pada pembentukan model dan pelabelan pada saat pelatihan.



Gambar 5.5 Peta topografi PCA + GSOM pada percobaan ke 2.

5.5.4 Pengetahuan baru dengan adanya visualisasi

Dengan adanya visualisasi model yang terbentuk pada proses pelatihan, peneliti dapat membuat hipotesis awal apakah model yang dibentuk mampu melakukan klasifikasi dengan baik atau tidak. Dari kasus diatas model yang dibentuk oleh GSOM tidak akan bisa mendeteksi serangan u2r. Jika pada data uji terdapat banyak serangan dengan label u2r maka tentu akurasi hasil klasifikasi akan menurun, sehingga peneliti dapat menentukan langkah untuk memperbaiki modelnya.

Dengan adanya visualisasi model, peneliti juga dapat mengetahui bahwa penggunaan metode CRV pada RGSOM saat proses pelatihan memungkinkan terdapat semua label atau kelas yang terdapat pada data latih. Karena pengukuran jarak terdekat setiap input pada data latih dengan node (*reference vector*) hanya dilakukan pada node kosong atau yang memiliki label yang sama, dengan ketentuan semua label atau kelas berada pada awal urutan input data latih, atau

masih terdapat node yang belum diberi label. Dari pengujian, penggunaan metode CRV mampu memberikan kontribusi untuk melakukan deteksi terhadap kelas yang memiliki distribusi data yang sedikit atau data latih yang bersifat *unbalance*.

5.6 Kemampuan metode RGSOM dalam deteksi intrusi jaringan

Berdasarkan pada Gambar 5.1 dan Gambar 5.2, yang menunjukkan tentang perbandingan waktu komputasi dan akurasi antara metode RGSOM dengan beberapa metode pembanding. Kecepatan waktu komputasi metode RGSOM baik menggunakan ataupun tidak menggunakan reduksi fitur menunjukkan waktu komputasi yang paling efisien dibandingkan dengan metode pembanding lainnya. Sedangkan akurasi, *precision*, dan *recall* yang dihasilkan oleh metode RGSOM berada diatas 99 %.

Beberapa hal membuat metode RGSOM memiliki kemampuan yang optimal dalam waktu komputasi dan akurasi antara lain, bahasa pemrograman yang digunakan, karakteristik dari metode RGSOM, dan penentuan parameter. Pada penelitian sebelumnya yang menggunakan bahasa pemrograman javascript, memerlukan waktu komputasi sebesar 19.892 detik pada saat pembelajaran dan 6.299 detik pada saat pengujian (Christyawan et al., 2019). Penggunaan bahasa python yang memiliki keunggulan dalam manipulasi array juga menjadi salah satu yang mempengaruhi waktu komputasi RGSOM pada penelitian tesis ini. Kemampuan metode RGSOM tersebut ditunjang dengan penentuan parameter yang tepat dan karakteristik dari metode RGSOM dimana peta topografi mampu berkembang secara adaptif. Beberapa hal pendukung tersebut yang membuat proses klasifikasi dengan metode RGSOM pada sistem deteksi intrusi jaringan memiliki kemampuan untuk melakukan klasifikasi dengan waktu komputasi yang efisien tanpa mengorbankan akurasi.

BAB 6 PENUTUP

6.1 Kesimpulan

Kesimpulan yang dapat diambil dari penelitian ini adalah:

1. Nilai parameter RGSOM yang di peroleh dengan menggunakan metode K-Fold untuk mencapai hasil yang optimal adalah *learning rate* awal (LRStart) adalah 1, *learning rate* akhir (LRStop) sebesar 6.00E-6, Radius adalah 1, maximum epoch adalah 3, dan *Clustering Treshold* (CT) sebesar 0.5.
2. Perbandingan waktu komputasi RGSOM dibandingkan dengan SOM, GSOM, dan SVM menunjukkan bahwa RGSOM mampu memberikan efisiensi waktu komputasi terbaik, pada proses pelatihan dengan rata-rata sebesar 519.4 detik dan proses pengujian sebesar 884.8 detik.
3. Perbandingan nilai akurasi yang dihasilkan menggunakan metode RGSOM dengan SOM, GSOM, dan SVM menduduki peringkat ke tiga, dan metode SVM merupakan yang tertinggi. Walaupun menduduki peringkat ke tiga, akurasi RGSOM sebesar 99,53%, *Detection Rate* (DTR) sebesar 99.90%, *precision* senilai 99.52%, dan *false alarm rate* (FAR) senilai 1.93%. Jika dibandingkan dengan akurasi metode SVM dengan reduksi fitur (PCA) yang memiliki akurasi tertinggi, metode RGSOM memiliki selisih lebih rendah 0.30% dalam hal akurasi. Metode RGSOM juga mampu mengatasi permasalahan data yang bersifat *unbalance*.
4. Reduksi fitur dengan menggunakan Principle Component Analysis (PCA) mampu mempercepat waktu komputasi baik ketika diterapkan pada SVM maupun RGSOM sehingga metode RGSOM lebih effisien.
5. Visualisasi peta topografi pada RGSOM maupun SOM dan GSOM, mampu memberikan pengetahuan baru pada proses pembelajaran, dalam penelitian ini mampu menunjukkan apakah model yang terbentuk akan mampu mendeteksi sebuah kelas tertentu yang jumlahnya sangat sedikit, dalam penelitian ini adalah jenis serangan U2R.
6. Penelitian pada tesis ini mampu membuktikan bahwa RGSOM mampu melakukan deteksi intrusi jaringan dengan waktu komputasi yang efisien tanpa mengorbankan akurasi.

6.2 Saran

Penentuan parameternya pada penelitian ini masih menggunakan metode bertahap untuk menentukan parameter terbaik. Kemungkinan solusi lain untuk menentukan parameter adalah melakukan penelitian menggunakan metode *meta heuristic* dalam penentuan parameter. Metode RGSOM yang terbukti mampu memberikan efisiensi waktu komputasi tanpa mengorbankan akurasi pada deteksi intrusi jaringan dengan *data set* KDD 1999 dapat menjadi acuan untuk melakukan penelitian di permasalahan lain dengan *data set* yang berbeda.

DAFTAR PUSTAKA

Agrawal, S., & Agrawal, J. (2015). Survey on Anomaly Detection using Data Mining Techniques. *Procedia Computer Science*, *60*, 708–713. https://doi.org/10.1016/j.procs.2015.08.220

Akinduko, A. A., Mirkes, E. M., & Gorban, A. N. (2016). SOM: Stochastic initialization versus principal components. *Information Sciences*, *364–365*, 213–221. https://doi.org/10.1016/j.ins.2015.10.013

Al Huda, F., Firdaus Mahmudy, W., & Tolle, H. (2016). Android Malware Detection Using Backpropagation Neural Network. *Indonesian Journal of Electrical Engineering and Computer Science*, *4*(1), 240. https://doi.org/10.11591/ijeecs.v4.i1.pp240-244

Alahakoon, D., Halgamuge, S. K., & Srinivasan, B. (2000). Dynamic self-organizing maps with controlled growth for knowledge discovery. *IEEE Transactions on Neural Networks*, *11*(3), 601–614. https://doi.org/10.1109/72.846732

Almalawi, A., Yu, X., Tari, Z., Fahad, A., & Khalil, I. (2014). An unsupervised anomaly-based detection approach for integrity attacks on SCADA systems. *Computers & Security*, *46*, 94–110. https://doi.org/10.1016/j.cose.2014.07.005

Almeida, V. A. F., Doneda, D., & de Souza Abreu, J. (2017). Cyberwarfare and Digital Governance. *IEEE Internet Computing*, *21*(2), 68–71. https://doi.org/10.1109/MIC.2017.23

Al-Yaseen, W. L., Othman, Z. A., & Nazri, M. Z. A. (2017). Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Systems with Applications*, *67*, 296–303. https://doi.org/10.1016/j.eswa.2016.09.041

Amato, F., Mazzocca, N., Moscato, F., & Vivenzio, E. (2017). Multilayer Perceptron: An Intelligent Model for Classification and Intrusion Detection. *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 686–691. https://doi.org/10.1109/WAINA.2017.134

Checkpoint. (2018). Live Cyber Attack Threat Map. Retrieved September 1, 2018, from https://threatmap.checkpoint.com/ThreatPortal/livemap.html

Chiba, Z., Abghour, N., Moussaid, K., El Omri, A., & Rida, M. (2018). A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection. *Computers & Security*, *75*, 36–58. https://doi.org/10.1016/j.cose.2018.01.023

Christyawan, T. Y., Supianto, A. A., & Mahmudy, W. F. (2019). Anomaly-based intrusion detector system using restricted growing self organizing map. *Indonesian Journal of Electrical Engineering and Computer Science*, *13*(3), 919–926. https://doi.org/10.11591/ijeecs.v13.i3.pp919-926

De la Hoz, E., De la Hoz, E., Ortiz, A., Ortega, J., & Martínez-Álvarez, A. (2014). Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps. *Knowledge-Based Systems*, *71*, 322–338. https://doi.org/10.1016/j.knosys.2014.08.013

Hamamoto, A. H., Carvalho, L. F., Sampaio, L. D. H., Abrão, T., & Proença, M. L. (2018). Network Anomaly Detection System using Genetic Algorithm and Fuzzy Logic. *Expert Systems with Applications*, *92*, 390–402. https://doi.org/10.1016/j.eswa.2017.09.013

Han, X., Xu, L., Ren, M., & Gu, W. (2015). A Naive Bayesian Network Intrusion Detection Algorithm Based on Principal Component Analysis. *2015 7th International Conference on Information Technology in Medicine and Education (ITME)*, 325–328. https://doi.org/10.1109/ITME.2015.29

Hosseini Bamakan, S. M., Wang, H., Yingjie, T., & Shi, Y. (2016). An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. *Neurocomputing*, *199*, 90–102. https://doi.org/10.1016/j.neucom.2016.03.031

Ikram, S. T. (2016). Improving Accuracy of Intrusion Detection Model Using PCA and optimized SVM. *Journal of Computing and Information Technology*, *24*(2), 133–148. https://doi.org/10.20532/cit.2016.1002701

Jabez, J., & Muthukumar, B. (2015). Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach. *Procedia Computer Science*, *48*, 338–346. https://doi.org/10.1016/j.procs.2015.04.191

Jayalakshmi, T., & Santhakumaran, A. (2011). Statistical Normalization and Back Propagation for Classification. *International Journal of Computer Theory and Engineering*, 89–93. https://doi.org/10.7763/IJCTE.2011.V3.288

Kabir, E., Hu, J., Wang, H., & Zhuo, G. (2018). A novel statistical technique for intrusion detection systems. *Future Generation Computer Systems*, *79*, 303–318. https://doi.org/10.1016/j.future.2017.01.029

Karami, A., & Guerrero-Zapata, M. (2015). A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks. *Neurocomputing*, *149*, 1253–1269. https://doi.org/10.1016/j.neucom.2014.08.070

Kim, G., Lee, S., & Kim, S. (2014). A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications*, *41*(4), 1690–1700. https://doi.org/10.1016/j.eswa.2013.08.066

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks*, *37*, 52–65. https://doi.org/10.1016/j.neunet.2012.09.018

Lin, W.-C., Ke, S.-W., & Tsai, C.-F. (2015). CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-Based Systems*, *78*, 13–21. https://doi.org/10.1016/j.knosys.2015.01.009

M. Ibrahim, L., T. Basheer, D., & S. Mahmod, M. (2013). *A comparison study for intrusion database (KDD99, NSL-KDD) based on self organization map (SOM) artificial neural network* (Vol. 8).

Murty, M. N., & Raghava, R. (2016). *Support Vector Machines and Perceptrons: Learning, Optimization, Classification, and Application to Social Networks*. Retrieved from http://dx.doi.org/10.1007/978-3-319-41063-0

Ruan, Z., Miao, Y., Pan, L., Patterson, N., & Zhang, J. (2017). Visualization of big data security: a case study on the KDD99 cup data set. *Digital Communications and Networks*, *3*(4), 250–259. https://doi.org/10.1016/j.dcan.2017.07.004

Skottler, S. (2018). *February 28th DDoS Incident Report*. Retrieved from Github website: https://githubengineering.com/ddos-incident-report/

Stolfo, S. J., Fan, W., Lee, W., Prodromidis, A., & Chan, P. K. (2000). *Cost-based Modeling and Evaluation for Data Mining With Application to Fraud and Intrusion Detection: Results from the JAM Project*.

UCI. (1999). *KDD Cup 1999 Task Description*. Retrieved from https://kdd.ics.uci.edu/databases/kddcup99/task.html

Vaarandi, R., & Podins, K. (2010). Network IDS alert classification with frequent itemset mining and data clustering. *2010 International Conference on Network and Service Management*, 451–456. https://doi.org/10.1109/CNSM.2010.5691262

Wang, Z., & Xue, X. (2014). Multi-Class Support Vector Machine. In Y. Ma & G. Guo (Eds.), *Support Vector Machines Applications* (pp. 23–48). https://doi.org/10.1007/978-3-319-02300-7_2

www.lohninger.com. (2012, October 20). Kohonen Network - Background Information. Retrieved October 9, 2018, from http://www.lohninger.com/helpcsuite/kohonen_network_-_background_information.htm

Zargar, S. T., Joshi, J., & Tipper, D. (2013). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE Communications Surveys & Tutorials*, *15*(4), 2046–2069. https://doi.org/10.1109/SURV.2013.031413.00127

LAMPIRAN A CONTOH DATA SET KDD 1999

Fitur yang terdapat pada *data set* KDD 1999 adalah sebagai berikut:

duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, is_host_login, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate, class

Contoh data dari kelas yang ada yang telah dilakukan pemetaan kelas:

[[0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.0,0.0,0.0,0.0,0.0,
1.0,0.0,0.0,255,255,1.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,dos],
[0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.0,0.0,0.0,0.0,0.0,1.0
,0.0,0.0,255,243,0.95,0.01,0.0,0.0,0.0,0.0,0.0,0.0,0.0,normal],
[0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0.0,0.0,1.0,1.0,0.5,1.0,
0.0,255,1,0.0,1.0,1.0,0.0,0.0,0.0,0.0,1.0,1.0,probe],
[1,tcp,ftp,SF,1262,2451,0,0,0,28,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.0,0.0,0.0,0.0,0.0,1.0,
0.0,0.0,255,61,0.24,0.02,0.0,0.0,0.0,0.0,0.0,0.0,0.0,r2l],
[127,tcp,telnet,SF,1567,2736,0,0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,83,1,0.99,0.0,0.0,0
.0,0.01,0.08,0.0,5,5,1.0,0.0,0.2,0.0,0.0,0.0,0.0,0.0,0.0,u2r]]

Contoh data yang telah dilakukan label encoder:

[[0,0,15,9,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.0,0.0,0.0,0.0,0.0,1.0,0.0
,0.0,255,255,1.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,dos],
[0,2,49,9,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,
255,243,0.95,0.01,0.0,0.0,0.0,0.0,0.0,0.0,0.0,normal],
[0,1,49,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0.0,0.0,1.0,1.0,0.5,1.0,0.0,255,1
,0.0,1.0,1.0,0.0,0.0,0.0,1.0,1.0,probe],
[1,1,19,9,1262,2451,0,0,0,28,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.0,0.0,0.0,0.0,0.0,1.0,0.0,
0.0,255,61,0.24,0.02,0.0,0.0,0.0,0.0,0.0,0.0,0.0,r2l],
[127,1,60,9,1567,2736,0,0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,83,1,0.99,0.0,0.0,0.0,0.0,0.01
,0.08,0.0,5,5,1.0,0.0,0.2,0.0,0.0,0.0,0.0,0.0,0.0,u2r]]

Contoh data yang telah dilakukan normalisasi degan penskalaan minmax:

[[0.0,0.0,0.21739130434782608,0.9,1.4883707192251517e-06,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,1.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,
0.0,1.0,1.0,1.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,dos],

[0.0,1.0,0.7101449275362319,0.9,1.5143306736302417e-07,
2.8319446459564872e-05,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0019569471624266144,0.0019569471624266144,0.0,0.0,0.0,0.0,
1.0,0.0,0.0,1.0,0.9529411764705882,0.95,0.01,0.0,0.0,0.0,0.0,0.0,0.0,normal],
[0.0,0.5,0.7101449275362319,0.1,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.003913894324853229,0.0019569471624266144,0.0
,0.0,1.0,1.0,0.5,1.0,0.0,1.0,0.00392156862745098,0.0,1.0,1.0,0.0,0.0,0.0,1.0,1.
0,probe],
[1.7144130706852507e-05,0.5,0.2753623188405797,0.9,
1.8200812477346334e-06,0.000475417556602295,0.0,0.0,0.0,
0.9333333333333333,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0019569
471624266144,0.0019569471624266144,0.0,0.0,0.0,0.0,1.0,0.0,0.0,1.0,0.2392
156862745098,0.24,0.02,0.0,0.0,0.0,0.0,0.0,0.0,r2l],
[0.0021773045997702683,0.5,0.8695652173913043,0.9,2.259958252931989
2e-06,0.000530698667899791,0.0,0.0,0.0,0.0333333333333333,0.0,1.0,0.0,
0.0,0.0,0.0,0.03571428571428571,0.0,0.0,0.0,0.0,0.0,0.162426614481409,0.0
019569471624266144,0.99,0.0,0.0,0.0,0.01,0.08,0.0,0.0196078431372549,0.0
196078431372549,1.0,0.0,0.2,0.0,0.0,0.0,0.0,0.0,u2r]]

LAMPIRAN B PRINCIPLE COMPONENT ANALYSIS

Principle Component yang terbentuk dengan 95% varian data latih.

PC1:

[3.96500529e-04, 1.88170200e-01, 1.65557755e-01, -1.46714350e-01, 2.36805036e-06, 3.50402249e-05, 4.03214509e-05, 1.04810051e-03, 2.59470952e-06, 4.61662372e-04, 1.86034114e-05, 2.71200625e-02, 5.05953209e-06, 3.50287618e-05, 1.15065599e-05, 6.01856835e-06, 1.89858247e-05, 1.94045962e-05, 3.24463568e-05, 0.00000000e+00, 0.00000000e+00, 6.82576778e-04, -2.42485997e-01, -3.53536380e-01, 2.74228133e-01, 2.74214458e-01, 4.70393419e-02, 4.69805914e-02, -2.96269435e-01, 2.54390826e-02, 6.68256060e-03, -1.76586845e-02, -3.23082511e-01, -3.19359541e-01, 3.02908385e-02, -3.54209138e-01, 1.14127861e-03, 2.74228301e-01, 2.74154908e-01, 4.72027806e-02, 4.68290244e-02]

PC2:

[9.43997058e-04, 2.02682299e-01, 1.47903799e-02, 1.98471867e-02, 2.81126661e-06, 4.26554848e-04, -8.77530813e-06, 2.95745119e-03, 1.22866899e-05, 2.93837535e-03, 7.25021873e-05, 4.18475678e-01, 3.06029098e-05, 2.97038861e-04, 4.45208942e-05, 2.88506544e-05, 9.74881808e-05, 1.30365445e-04, 3.41847936e-04, -0.00000000e+00, -0.00000000e+00, 3.46796415e-03, -4.09285344e-01, -3.37612169e-01, -2.54027553e-01, -2.54416295e-01, 1.03550970e-01, 1.04868555e-01, 1.70532741e-01, -4.04055277e-03, 7.90705567e-02, -2.49064370e-01, 8.25550972e-02, 1.16914780e-01, 8.46334111e-03, -2.85103708e-01, 1.68715487e-02, -2.53720862e-01, -2.54701878e-01, 1.05056861e-01, 1.04346712e-01],

PC3:

[9.30944714e-04, -4.06009084e-02, 4.18151565e-02, -2.50751445e-01, 1.06833757e-06, -2.66925514e-04, -8.85334598e-05, 7.65806826e-04, -4.81042756e-06, -1.47992612e-03, 1.03259950e-04, -2.92821012e-01, -1.62367069e-05, -1.61739274e-04, -2.12254664e-05, -1.35926233e-05, -4.67401866e-05, -6.59260402e-05, -2.24878868e-04, -0.00000000e+00, -0.00000000e+00, -1.61795677e-03, 1.38064475e-01, 6.54549444e-02, -1.57271189e-01, -1.58736532e-01, 3.95942819e-01, 3.96073134e-01, -1.65723242e-01, 3.38208251e-02, -3.33258506e-02, 9.44619271e-02, -1.65646902e-01, -1.69656617e-01, 4.58103297e-02, 8.34691555e-02, 1.75254304e-03, -1.57366248e-01, -1.58246233e-01, 3.92148740e-01, 3.91834806e-01],

PC4:

[2.15914286e-02, 3.22207360e-01, 1.12037113e-01, 2.05923304e-01, 1.23360190e-05, 2.59483092e-04, -6.75546129e-05, 3.77761529e-02,

5.58926085e-05, 6.00867683e-03, -1.89302591e-05, -2.55380330e-01, 4.54399144e-05, 4.04393652e-04, 1.35210544e-04, 9.36626640e-05, 2.65637288e-04, 4.27828572e-04, 8.96069694e-05, 0.00000000e+00, 0.00000000e+00, 1.10207759e-02, -1.85807472e-01, -9.64114629e-02, -1.70534062e-01, -1.72733366e-01, -1.62183783e-01, -1.62649552e-01, 1.50302076e-01, 6.49856720e-02, -2.08170671e-03, 2.12461531e-01, -3.84956873e-01, -3.50026247e-01, 2.94509818e-01, 2.89528821e-01, -1.54115746e-02, -1.69183191e-01, -1.70996921e-01, -1.49471592e-01, -1.57432970e-01],

PC5:

[4.88540697e-03, -1.26296917e-01, -5.93641623e-02, -4.28047889e-02, 2.09272833e-05, 7.22059931e-04, 1.55421493e-03, -2.62282409e-03, 5.85734205e-05,-3.02273962e-03, 3.90066276e-04, -8.10352582e-02, 1.68716183e-05, 9.33617522e-04, 4.67266172e-05, 8.21260687e-05, 2.08853687e-04, 4.25007063e-04,-4.51894478e-04, 0.00000000e+00, 0.00000000e+00,-2.89311893e-05, -6.41027931e-02, -1.53201766e-02, 7.05834780e-02, 7.17024698e-02, 2.51857729e-02, 2.01949583e-02, 9.75577174e-02, -4.72611683e-03, 1.80264489e-01, -8.10289208e-01, -2.37994206e-01,-2.89844473e-02, 7.10955857e-02, 4.17788234e-01, 1.00748443e-01, 7.05971213e-02, 7.01659232e-02, 2.36903559e-02, 1.63424332e-02]

Contoh data yang telah mengalami reduksi fitur dengan metode PCA:

[[PC1,PC2,PC3,PC4,PC5,class],
[-0.8987102300285871,-0.2933076351385662,0.03548367828693436,-0.010744263701359629,-0.000406665262114888,dos],
[0.3515790526468409,0.9375562199538294,-0.2543668830120345, 0.39736965153807585,-0.48111625260107344,normal],
[1.0217984328737078,0.6819120857765384,2.1066841908158747,0.7082105399526905,0.389628371805351,probe],
[0.6713862734978482,1.112624265658076,-0.3089265501332798, 0.4750204804747801,-0.28488985050684695,r2l],
[1.0706836999903608,0.8867062087240017,-0.41729506477768186,-0.1547850202097947,0.5520082345948265,u2r]]

LAMPIRAN C PCA INITIALIZATION

PCA Initialization
Pembentukan map 50x50 dengan metode PCA Initialization
Tahap 1 (41 fitur):

```
[[[-1.34049759e-03, -3.90852498e-01, -1.80348134e-01, ..., -1.51175736e-01],
  [-1.30273770e-03, -3.82745206e-01, -1.79756519e-01, ..., -1.47001868e-01],
  [-1.26497782e-03, -3.74637914e-01, -1.79164904e-01, ..., -1.42827999e-01],
  ...,
  [ 1.21135792e-03, 3.59003815e-01, 1.71950979e-01, ..., 1.36780970e-01],
  [ 1.24911780e-03, 3.67111106e-01, 1.72542594e-01, ..., 1.40954838e-01],
  [ 1.28687768e-03, 3.75218398e-01, 1.73134209e-01, ..., 1.45128707e-01]]]
```

Tahap 2 (5 fitur):

```
[[[ 0.10394356, -0.40332243, 0.37164599, 0.1444785, 0.69236778],
  [ 0.10394356, -0.36332243, 0.37164599, 0.1444785, 0.69236778],
  [ 0.10394356, -0.32332243, 0.37164599, 0.1444785, 0.69236778],
  ...,
  [ 2.06394356, 1.47667757, 0.37164599, 0.1444785, 0.69236778],
  [ 2.06394356, 1.51667757, 0.37164599, 0.1444785, 0.69236778],
  [ 2.06394356, 1.55667757, 0.37164599, 0.1444785, 0.69236778]]]
```

Data lengkap yang belum dan telah melalui tahap preprocessing dapat di unduh pada url berikut: http://blog.ub.ac.id/tomiyahya/file-pendukung-kdd-1999/