TESIS KI142502

PENGEMBANGAN METODE SELEKSI FITUR DAN TRANFORMASI DATA PADA SISTEM DETEKSI INTRUSI DENGAN PEMBATASAN UKURAN *CLUSTER* DAN SUB-MEDOID

Indera Zainul Mutaqien
5114201034

DOSEN PEMBIMBING
Tohari Ahmad, S.Kom., MIT., Ph.D.

PROGRAM MAGISTER
BIDANG KEAHLIAN KOMPUTASI BERBASIS JARINGAN
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

[Halaman ini sengaja dikosongkan]

TESIS KI142502

FEATURE SELECTION AND DATA TRANSFORMATION ON INTRUSION DETECTION SYSTEM USING CLUSTER SIZE AS THRESHOLD AND SUB-MEDOID

Indera Zainul Mutaqien
5114201034

SUPERVISOR
Tohari Ahmad, S.Kom., MIT., Ph.D.

MASTER PROGRAMME
FIELD OF EXPERTISE IN NETWORK CENTRIC COMPUTING
DEPARTMENT OF INFORMATICS ENGINEERING
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

[Halaman ini sengaja dikosongkan]

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar

Magister Komputer (M.Kom.)

di

Institut Teknologi Sepuluh Nopember Surabaya

oleh :

INDERA ZAINUL MUTTAQIEN

NRP. 5114201034

Dengan judul :

Pengembangan Metode Seleksi Fitur dan Transformasi Data pada Sistem Deteksi Intrusi Dengan Pembatasan Ukuran *Cluster* dan Sub-medoid

Tanggal Ujian : 21-11-2016

Periode Wisuda : 2016 Ganjil

Disetujui oleh :

Tohari Ahmad, S.Kom., MIT., Ph.D. (Pembimbing)
NIP.197505252003121002

Royyana Muslim I, S.Kom., M.Kom., Ph.D. (Penguji 1)
NIP.197708242006041001

Dr.Eng. Radityo Anggoro, S.Kom, M.Sc. (Penguji 2)
NIP.198410162008121002

Hudan Studiawan, S.Kom., M.Kom. (Penguji 3)
NIP.198705112012121003

Direktur Program Pasca Sarjana,

an. Direktur Program Pascasarjana
Asisten Direktur

Prof. Dr. Ir. Tri Widjaja, M.Eng.
NIP. 19611021 198603 1 001

Prof. Ir. Djauhar Manfaat, M.Sc., Ph.D.
NIP. 19601201987011001

[Halaman ini sengaja dikosongkan]

Pengembangan Metode Seleksi Fitur dan Transformasi Data pada Sistem Deteksi Intrusi Dengan Pembatasan Ukuran *Cluster* dan Sub-medoid

Nama Mahasiswa	: Indera Zainul Muttaqien
NRP	: 5114201034
Pembimbing	: Tohari Ahmad, S.Kom., MIT., Ph.D.

ABSTRAK

Penanganan keamanan jaringan mutlak diperlukan supaya data dalam jaringan tetap terjaga dari serangan. Sistem deteksi intrusi / *Intrusion Detection System* (IDS) muncul sebagai salah satu solusi untuk menangani hal tersebut. Beberapa penelitian terdahulu menunjukkan penggunakan teknik *machine learning* untuk mendeteksi intrusi dapat memberikan nilai *accuracy* yang baik.

Teknik *machine learning* ini tidak terlepas dari proses seleksi fitur untuk mengoptimalkan pemrosesan oleh algoritma *learning*. Proses seleksi fitur dapat dilakukan untuk menghindari resiko *overfit* dan meningkatkan akurasi proses deteksi. Pemrosesan juga dapat dilakukan lebih cepat karena berkurangnya dimensi.

Pengurangan dimensi dari suatu *dataset* dapat dilakukan dengan seleksi fitur dan transformasi data. Sejumlah penelitian telah dilakukan untuk melakukan transformasi *dataset* ke satu dimensi dengan menggunakan pendekatan metode *centroid-based*. Metode ini melakukan transformasi data ke satu dimensi dengan memanfaatkan jarak data ke *centroid* dari suatu *dataset* sebagai pembeda antar data. Namun demikian masih tersedia ruang untuk meningkatkan hasil dari penelitian-penelitian tersebut.

Pada penelitian ini diajukan suatu sistem deteksi intrusi yang terdiri dari serangkaian proses seleksi fitur, *clustering*, dan transformasi data dengan pendekatan metode *centroid based*. Proses seleksi fitur dilakukan secara bertahap dengan menggabungkan teknik *filter* dan *wrapper* untuk memperoleh fitur-fitur yang tepat. Sistem ini juga menggunakan nilai yang disebut sebagai ambang radius untuk membatasi ukuran *cluster* yang terbentuk pada proses *clustering*. Proses transformasi data dilakukan dengan memanfaatkan jarak data ke *centroid* dan jarak data ke beberapa *sub-medoid* untuk meningkatkan akurasi hasil deteksi.

Hasil penelitian ini menunjukkan bahwa pemilihan fitur-fitur yang tepat (signifikan) pada *dataset* dapat meningkatkan performa hasil deteksi. Pada dataset NSLKDD yang digunakan pada penelitian ini ditemukan ada 19 fitur signifikan, sedangkan pada dataset Kyoto2006+ terdapat 14 fitur signifikan. Selain itu metode yang diajukan secara umum memberikan perbaikan hasil deteksi pada setiap dataset yang diuji. Hasil terbaik terlihat pada penerapan metode yang diajukan pada dataset Kyoto2006+.

Kata kunci : ambang radius, deteksi intrusi, seleksi fitur, *sub-medoid*.

[Halaman ini sengaja dikosongkan]

Feature Selection and Data Transformation on Intrusion Detection System Using Cluster Size as Threshold and Sub-medoid

Name	: Indera Zainul Muttaqien
Student Identity Number	: 5114201034
Supervisor	: Tohari Ahmad, S.Kom., MIT., Ph.D.

ABSTRACT

In managing network security, it is absolutely necessary do protect the data within the network from any attack / malicious access. Intrusion Detection System (IDS) has emerged as one of the solutions to deal with problem. Previous studides have shown that the use machines learning techniques to detect intrusion provide better performance in term of accuracy.

The application of machine learning techniques can not be separated from the feature selection process to optimize the processing by learning algorithms. The usage of feature selection has advantages such as to avoid overfit risks and to improve detection accuracy. The processing time is also decrease due to the reduced dimensions.

The dimensionality reduction of a dataset can be done by the selection of significant features and data transformation. Numerous studies have been done to transform dataset features into a one-dimensional form using centroid-based approach. This approach performs the transformation by using the distance between data and clusters centroid of a dataset as a differentiator between data. However, there is room to improve the outcome of these studies.

This study proposed an intrusion detection system which comprises of a series of feature selection process, clustering, and data transformation with the centroid-based approach. The feature selection process carried out gradually by combining the filter technique and the wrapper technique to obtain the significant feature(s). The proposed system is also used a value called radius threshold to limit the size of the clusters formed in the clustering process. Data transformation process is done by summing the distance between data and the centroids; and the distance between data and a number of sub-medoids. This is intended to improve the accuracy of the detection.

The results of this study indicate that the selection of appropriate features of the dataset can improve the detection performance. It is found that the NSLKDD used in this study has 19 significant features, while the dataset Kyoto2006+ has 14 significant features. In addition, the proposed method generally able to provide improvements to the detection results in each dataset tested. The best results gained in the application of the method proposed in the dataset Kyoto2006+.

Keywords : feature selection, intrusion detection, radius threshold, sub-medoid.

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Segala puji syukur bagi Allah SWT, Sang pencipta alam semesta, Yang Maha Mengetahui, dan Penguasa semua ilmu pengetahuan. Hanya karena rahmat, hidayah, dan inayah-Nya buku tesis ini dapat diselesaikan dengan baik. Shalawat dan salam senantiasa tersampaikan kepada Rasulullah Muhammad SAW dan keluarga, para sahabat, dan para pengikutnya.

Buku tesis ini disusun sebagai salah satu syarat memperoleh gelar Magister Komputer pada program magister Teknik Informatika di Institut Teknologi Sepuluh Nopember, dengan judul "Pengembangan Metode Seleksi Fitur dan Transformasi Data pada Sistem Deteksi Intrusi Dengan Pembatasan Ukuran *Cluster* dan *Sub-medoid*".

Penulis menyadari sepenuhnya bahwa banyak pihak yang telah membantu dalam penyelesaian tesis ini. Untuk itu dengan segala kerendahan hati, penulis menyampaikan terima kasih yang sedalam-dalamnya kepada :

1. Alm. Ayahanda Achmad Wasil dan Ibunda Maimunah Maschab tercinta serta kakak Rani Fitria Anggraini atas kasih sayang, doa, dan dukungannya.
2. Istriku tercinta Umi Muniroh, atas segala doa, perhatian, pengertian, dan motivasi tiada hentinya untuk segera menyelesaikan studi pascasarjana.
3. Bapak Dr. Waskitho Wibisono, S.Kom., M.Eng. selaku Ketua Program Magister Teknik Informatika yang telah memberi dukungan dan arahan dalam menyelesaikan permasalahan akademik.
4. Bapak Tohari Ahmad, S.Kom., MIT., Ph.D. selaku dosen pembimbing yang telah dengan sabar membimbing, memberikan ilmu, meluangkan waktu dan pikiran dalam proses pengerjaan tesis ini.
5. Bapak Dr. Royyana Muslim Ijtihadie, S.Kom., M.Kom., Ph.D., Bapak Dr.Eng. Radityo Anggoro, S.Kom., M.Sc., dan Bapak Hudan Studiawan S.Kom., M.Kom. selaku dosen penguji yang telah banyak memberikan motivasi dan saran yang mendukung terselesaikannya tesis ini.

6. Seluruh dosen beserta staf di S2 Teknik Informatika ITS yang telah memberikan wawasan, ilmu pengetahuan baru, dan bantuan bagi penulis selama menempuh masa studi pascasarjana.
7. Bapak Prof. Dr. Ir. Suprapto, DEA. selaku Koordinator Kopertis Wilayah VII, Bapak Prof. Dr. Ali Maksum selaku Sekretaris Pelaksana Kopertis Wilayah VII, Bapak Drs.Ec. Purwo Bekti, M.Si. selaku Kepala Bidang Kelembagaan dan Sistem Informasi, dan Bapak Drs. Supradono, MM. selaku Kepala Seksi Sistem Informasi, atas izin, fasilitas, dan bantuannya selama penulis melaksanakan tugas belajar, serta segenap pejabat struktural dan rekan-rekan di Kopertis Wilayah VII yang tidak dapat penulis sebutkan satu persatu.
8. Rekan-rekan di S-2 Teknik Informatika angkatan 2014, khususnya Mas Muhammad Machmud, yang telah berbagi dan saling mendukung dalam masa perkuliahan sampai masa penyelesaian tesis.
9. Kepada semua pihak yang telah membantu yang tidak dapat penulis sebutkan satu persatu.

Akhir kata, begitu banyak kekurangan pada setiap hasil karya manusia, begitu pula pada tesis ini. Saran, masukan, dan kritik yang membangun selalu penulis harapkan demi perbaikan dan penerapan tesis ini di masa mendatang. Semoga tesis ini dapat memberikan manfaat dan kontribusi bagi pribadi penulis, masyarakat, bangsa, dan negara.

Surabaya, Desember 2016

Penulis,
Indera Zainul Muttaqien

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Pada era internet seperti saat ini, informasi dalam beragam bentuk dan media dapat diakses dengan cepat dan mudah. Teknologi jaringan yang digunakan untuk mengirimkan dan menerima data sangat rentan terhadap serangan. Penanganan keamanan jaringan mutlak diperlukan supaya data yang dipertukarkan ataupun disimpan di *host-host* dalam jaringan tetap terlindungi dari ancaman yang mungkin menyerang jaringan secara diam-diam. Salah satu solusi untuk menangani permasalahan ini adalah dengan memanfaatkan sistem deteksi intrusi / Intrusion Detection System (IDS). Secara umum IDS harus mampu membedakan akses normal ke jaringan atau akses serangan dan menyediakan informasi-informasi terkait serangan tersebut untuk analisa lebih lanjut. Menurut (Sommer dan Paxson, 2010), berdasarkan cara melakukan deteksi, IDS dibagi menjadi dua macam, yaitu IDS yang berbasis *misuse-detection* dan IDS yang berbasis deteksi anomali (*anomaly detection).*

IDS yang berbasis *misuse-detection* melakukan deteksi dengan menggunakan basis data pola serangan (*malicious behaviour*). Keunggulan sistem ini bergantung kepada ketersediaan pola serangan dalam basis data tersebut, karena itu sistem ini tidak mampu mendeteksi serangan baru yang memiliki perilaku berbeda sama sekali dengan data serangan yang dimilikinya.

IDS yang berbasis deteksi anomali menggunakan pendekatan pengenalan aktifitas normal. Setiap aktifitas jaringan yang memiliki pola berbeda dengan pola aktifitas normal akan dianggap sebagai anomali / serangan. Beberapa kelebihan dari IDS berbasis deteksi anomali adalah (Agrawal dan Agrawal, 2015): (i) mampu mendeteksi serangan dari dalam jaringan, misalnya pada saat terjadi akses oleh akun curian yang memiliki karakteristik akses berbeda dengan akses normal; (ii) memiliki model sistem yang spesifik pada jaringan tertentu sehingga menyulitkan penyerang dalam melakukan serangan karena karakteristik sistem yang tidak umum; dan (iii) dapat melakukan deteksi terhadap serangan baru yang belum

diketahui sebelumnya. IDS tipe ini menggunakan teknik *machine learning* untuk melakukan deteksi aktifitas.

Teknik *machine learning* melakukan pendeteksian dengan mencari kesamaan (*similarities*) dari suatu aktifitas akses jaringan berdasarkan sekumpulan data aktifitas yang sudah dikenali sebelumnya. Data aktifitas jaringan tersebut dapat terdiri dari ribuan sampai jutaan data dan masing-masing memiliki banyak fitur / atribut. Untuk mengoptimalkan pemrosesan oleh algoritma *learning*, digunakan pendekatan pengurangan dimensi / fitur (*dimensionality reduction*). Secara umum, pengurangan dimensi ini dilakukan dengan dua cara, yaitu : seleksi fitur dan ekstraksi/transformasi fitur.

Seleksi fitur dilakukan untuk menentukan fitur-fitur yang signifikan dalam *dataset* yang sesuai untuk permasalahan yang akan dipecahkan. Dengan menggunakan seleksi fitur, maka resiko *over fitting* dapat dihindari, akurasi menjadi lebih baik karena menghilangkan data redundan dan fitur-fitur yang tidak signifikan, dan dapat menghemat waktu yang dibutuhkan dalam proses klasifikasi karena berkurangnya dimensi.

Selain dengan seleksi fitur, pengurangan dimensi dari *dataset* dapat dilakukan dengan ekstraksi fitur / transformasi fitur. Ektraksi fitur dilakukan untuk mentransformasikan ruang dimensi dari fitur-fitur pada *dataset* asli ke ruang dimensi yang lebih sederhana. Sejumlah peneliti seperti (Tsai dan Lin, 2010), (Lin dkk., 2015), dan (Muchammad dan Ahmad, 2015) mengajukan metode transformasi data ke bentuk satu dimensi dengan memanfaatkan jarak data ke *centroid* (yang diperoleh dari proses *clustering*) dari suatu *dataset* sebagai pembeda antar data. Tsai dan Lin (2010) mengajukan metode TANN yang melakukan transformasi fitur untuk memperoleh fitur baru dengan menggunakan penjumlahan semua area segitiga yang dapat terbentuk antara data tersebut dengan setiap 2 centroid. Lin, dkk. (2015) menggunakan sejumlah *cluster* untuk memperoleh *centroid* dan menambahkan jarak data ke sejumlah tetangga terdekat (*nearest neighbors*) dalam satu *cluster*. Muchammad dan Ahmad (2015) menggunakan metode partisi dinamis dengan *threshold purity* dan entropi *cluster* dalam pengelompokan data ke suatu *cluster*. Metode mereka memberikan peningkatan

hasil dari penelitian sebelumnya dengan mengajukan penambahan jarak data ke sejumlah *sub-centroid* dalam satu *cluster*.

Pada penelitian ini, penulis menggunakan metode seleksi fitur dengan menggabungkan teknik *filter* dengan teknik *wrapper* (Guyon dan Elisseeff, 2003) untuk membangun suatu model deteksi intrusi. Penulis juga mengajukan penggunaan ukuran *cluster* sebagai *threshold* dalam proses *clustering* dan penggunaan jarak data ke *centroid* dan jarak data ke *sub-medoid cluster* untuk melakukan pembangkitan fitur transformasi pada model deteksi intrusi.

1.2 Perumusan Masalah

Beberapa permasalahan dalam penelitian ini dirumuskan sebagai berikut:

1. Bagaimana proses seleksi fitur terhadap *dataset* dilakukan ?
2. Apa saja fitur-fitur signifikan pada dataset berdasarkan proses seleksi fitur tersebut ?
3. Apakah pengaruh dari pembatasan ukuran *cluster* terhadap performa deteksi ?
4. Apakah penggunaan jarak ke *sub-medoid* memberikan pengaruh terhadap performa deteksi ?

1.3 Batasan Masalah

Dalam tesis ini, batasan masalah yang dibahas adalah sebagai berikut:

1. *Dataset* yang digunakan adalah NSL-KDD dan Kyoto2006+
2. Bahasa pemrograman yang digunakan adalah Java.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah memperoleh fitur-fitur signifikan dari suatu dataset sistem deteksi intrusi melalui proses seleksi fitur dengan metode *filter* dan metode *wrapper*. Penelitian ini juga bertujuan untuk menguji pengaruh dari pembatasan ukuran *cluster* dan penggunaan jarak ke *sub-medoid* terhadap performa deteksi.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini, yaitu : (i) memberikan subset fitur signifikan yang dari suatu *dataset* pengujian sistem deteksi intrusi dan (ii) memberikan alternatif metode untuk melakukan transformasi fitur pada model sistem deteksi intrusi.

1.6 Kontribusi Penelitian

Kontribusi dari penelitian ini adalah pengembangan seleksi fitur dengan menggabungkan teknik *filter* dan *wrapper*, penggunaan ambang radius untuk membatasi ukuran *cluster*, dan penggunaan jarak ke *sub-medoid* untuk membangkitkan fitur transformasi untuk model sistem deteksi intrusi.

BAB 2
KAJIAN PUSTAKA DAN DASAR TEORI

Dalam subbab ini diuraikan kajian pustaka dan dasar teori yang digunakan sebagai landasan ilmiah penelitian.

2.1 *Feature Selection* / Pemilihan Fitur

Pemilihan fitur adalah suatu proses yang dilakukan untuk menentukan fitur-fitur yang signifikan dalam *dataset* yang sesuai untuk permasalahan yang akan dipecahkan. Semakin baik hasil pemilihan fitur dapat meningkatkan nilai *accuracy* dari metode deteksi yang diuji. Pemilihan fitur juga bermanfaat dalam mereduksi dimensi dari *dataset* dengan cara 'membuang' fitur-fitur yang tidak signifikan (tidak memiliki pengaruh terhadap penentuan kelas / label). Tujuan utama dari seleksi fitur adalah memperoleh kumpulan fitur-fitur terbaik yang dapat meningkatkan performansi dari model deteksi yang dikembangkan.

Beberapa keuntungan dari seleksi fitur adalah :

a. Meminimalkan *overfit* : proses seleksi fitur dapat menghilangkan data redundan dan *noise* yang dapat mengakibatkan *overfit* pada proses clustering.
b. Meningkatkan *accuracy* : berkaitan dengan (a), proses seleksi fitur akan menghilangkan fitur-fitur yang tidak signifikan yang dapat mengakibatkan *misleading* akibat *overfit*, dengan demikian nilai *accuracy* akan meningkat.
c. Mengurangi waktu pemrosesan : semakin sederhana dimensi dari *dataset* maka algoritma *learning* dapat dijalankan dengan lebih cepat dan efisien.

Proses seleksi fitur antara lain melibatkan kombinasi dari proses pencarian, estimasi pengaruh fitur dalam penentuan label data, dan evaluasi dengan menggunakan algoritma *machine learning*. Dengan demikian seleksi fitur akan melibatkan banyak sekali kemungkinan proses (Hall dan Holmes, 2003). Untuk mengoptimalkan proses seleksi fitur, digunakan prosedur pencarian secara heuristik yang dipadukan dengan *evaluator* yang berfungsi untuk mengestimasi tingkat pengaruh fitur. Secara umum seleksi fitur dikelompokkan menjadi tiga teknik (Guyon dan Elisseeff, 2003), yaitu : teknik *filter*, teknik *wrapper*, dan teknik *embedded*.

Teknik *filter* menggunakan pengujian statistik untuk melakukan evaluasi terhadap fitur sehingga teknik ini tidak bergantung kepada algoritma *learning* tertentu. Teknik *filter* akan menghasilkan rangking fitur mulai dari fitur yang paling signifikan sampai yang tidak signifikan. Suatu fitur disebut tidak signifikan jika fitur tersebut tidak memiliki pengaruh dalam penentuan label dari data pada dataset.

Teknik *wrapper* melakukan seleksi fitur dengan membentuk *subset-subset* yang terdiri dari kombinasi yang mungkin dari fitur *dataset*. Kemudian masing-masing *subset* tersebut akan dievaluasi dengan algoritma *learning* untuk mendapatkan tingkat deteksi dalam penentuan label / kelas. Teknik *wrapper* dapat memberikan hasil yang lebih baik daripada teknik *filter*, namun membutuhkan tingkat komputasi yang lebih tinggi. Hasil dari teknik ini adalah *subset* fitur yang memberikan kontribusi paling baik.

Teknik *embedded* merupakan penggabungan keunggulan dari teknik *filter* yang cepat dan akurasi dari teknik *wrapper*. Pada teknik *embedded*, seleksi fitur dilakukan sebagai bagian dari proses *learning* terhadap seluruh data *training*, sehingga pada umumnya hasil seleksi fitur spesifik ke model yang dibangun.

2.2 *Dataset* NSL-KDD

NSL-KDD adalah *dataset* yang diajukan (Tavallaee dkk., 2009) sebagai solusi dari permasalahan yang ada pada *dataset* KDD Cup 1999 (KDD-99). *Dataset* KDD-99 (Hettich dan Bay, 1999) sendiri usianya sudah lebih dari 15 tahun, namun masih umum digunakan dalam penelitian-penelitian sistem deteksi intrusi karena tidak banyak *dataset* alternatif yang tersedia dan dapat diakses publik.

Beberapa masalah yang terdapat dalam *dataset* KDD-99 yang sudah ditangani pada NSL-KDD adalah penghapusan data redundan dan proporsi ulang terhadap *dataset*. NSL-KDD tidak menyertakan data redundan yang ada pada KDD-99 yang dapat mempengaruhi performa dari algoritma *learning*. Pada NSL-KDD, *dataset* asli (KDD-99) telah diproporsi ulang sehingga memungkinkan untuk digunakan dalah proses evaluasi berbagai macam algoritma *learning*.

NSL-KDD memuat seluruh fitur yang ada pada KDD-99 yang terdiri dari 41 fitur (Tabel 2.1) dan 23 kelas yaitu berupa 1 kelas normal dan 22 kelas tipe

serangan. Kelas-kelas tipe serangan tersebut kemudian dapat dikelompokkan menjadi 4 kelas (DoS, probe, R2L, dan U2R) seperti pada Tabel 2.2.

DoS (*Denial of Service*) adalah tipe serangan yang membebani sumber daya komputer (misalnya dengan *synflood* atau *ping of death*) sehingga komputer target mengalami *system crash* dan tidak mampu untuk memproses koneksi normal bahkan mengakibatkan user tidak dapat mengakses komputer tersebut.

R2L (r*emote to local*) adalah tipe serangan yang bertujuan untuk mendapatkan akses sebagai pengguna sistem. R2L dilakukan oleh penyerang yang memiliki akses ke sistem dan melakukan eksploitasi untuk mendapatkan akses lokal.

Serangan *Probe* bertujuan untuk mendapatkan informasi tentang status jaringan komputer dengan cara melakukan pemindaian terhadap komputer-komputer dalam jaringan tersebut. Informasi ini dapat digunakan oleh penyerang untuk memetakan jaringan yang berguna dalam melakukan penyerangan berikutnya.

U2R (*user to root*) adalah tipe serangan yang berusaha untuk mendapatkan akses root/admin pada komputer target dengan melakukan eksploitasi celah keamanan sistem. Serangan U2R umumnya dilakukan setelah penyerang mendapatkan akses user normal ke sistem (baik melalui *sniffing*, *social engineering*, ataupun *dictionary attack*).

Tabel 2.1 Daftar Fitur NSL-KDD

No urut	Nama Fitur	Tipe
1	duration	numerik
2	protocol_type	nominal
3	service	nominal
4	flag	nominal
5	src_bytes	numerik
6	dst_bytes	numerik
7	land	biner
8	wrong_fragment	numerik
9	urgent	numerik
10	hot	numerik

No urut	Nama Fitur	Tipe
11	num_failed_logins	numerik
12	logged_in	biner
13	num_compromised	numerik
14	root_shell	numerik
15	su_attempted	numerik
16	num_root	numerik
17	num_file_creations	numerik
18	num_shells	numerik
19	num_access_files	numerik
20	num_outbound_cmds	numerik
21	is_host_login	biner
22	is_guest_login	biner
23	count	numerik
24	srv_count	numerik
25	serror_rate	numerik
26	srv_serror_rate	numerik
27	rerror_rate	numerik
28	srv_rerror_rate	numerik
29	same_srv_rate	numerik
30	diff_srv_rate	numerik
31	srv_diff_host_rate	numerik
32	dst_host_count	numerik
33	dst_host_srv_count	numerik
34	dst_host_same_srv_rate	numerik
35	dst_host_diff_srv_rate	numerik
36	dst_host_same_src_port_rate	numerik
37	dst_host_srv_diff_host_rate	numerik
38	dst_host_serror_rate	numerik
39	dst_host_srv_serror_rate	numerik
40	dst_host_rerror_rate	numerik
41	dst_host_srv_rerror_rate	numerik

Tabel 2.2 Daftar Pengelompokan Kelas Serangan pada NSL-KDD

No urut	Nama Kelompok Kelas Serangan	Nama Kelas Tipe Serangan
1	DoS	back, land, neptune, pod, smurf, teardrop
2	R2L	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster

No urut	Nama Kelompok Kelas Serangan	Nama Kelas Tipe Serangan
3	Probe	ipsweep, nmap, portsweep, satan
4	U2R	buffer_overflow, loadmodule, perl, rootkit

Sumber : (Hettich dan Bay, 1999) dan (Singh dkk., 2015)

2.3 *Dataset* Kyoto2006+

Dataset Kyoto2006+ diajukan oleh (Song dkk., 2011) sebagai alternatif untuk *dataset* KDD-99 yang dipandang kurang dapat merefleksikan kondisi jaringan di dunia nyata. Song dkk. membangun sebuah *honeypot* untuk melakukan pengumpulan data pada periode tahun 2006 s.d. 2009. Akses serangan ke *honeypot* tersebut direkam untuk mendapatkan data serangan terhadap jaringan di dunia nyata. Sedangkan data akses normal diperoleh dengan menggunakan mail server dan DNS server yang melakukan akses normal ke *honeypot*. Beragam akses tersebut kemudian digunakan untuk membentuk *dataset* Kyoto2006+. Pelabelan data akses tersebut dilakukan oleh 3 perangkat lunak yaitu Symantec Network Security 7160 (SNS7160), ClamAV, dan Ashula.

Hasil dari penelitian tersebut adalah *dataset* yang memuat tipe-tipe serangan dan fitur-fitur baru yang dapat digunakan untuk penelitian dalam bidang deteksi intrusi pada jaringan. *Dataset* Kyoto2006+ terdiri dari 14 fitur yang diturunkan dari KDD-99 dan 10 fitur tambahan. Dalam penelitian ini akan digunakan 14 fitur saja dari Kyoto2006+ seperti pada Tabel 2.3 dan 1 fitur *label*. Fitur *label* merupakan kelas dari *dataset*, yang terdiri dari 3 nilai (Song dkk., 2011), yaitu ‘1’ jika data tersebut adalah normal, ‘-1’ jika merupakan data serangan yang dikenali oleh 3 perangkat lunak seperti di atas, dan ‘-2’ jika data merupakan data serangan yang tidak dikenali. *Dataset* terakhir dari Kyoto2006+ adalah data pada bulan Agustus 2009.

Tabel 2.3 Daftar Fitur Kyoto2006+

No urut	Nama Fitur	Tipe
1	duration	numerik
2	service	nominal
3	source bytes	numerik
4	destination bytes	numerik

No urut	Nama Fitur	Tipe
5	count	numerik
6	same_srv_rate	numerik
7	serror_rate	numerik
8	srv_serror_rate	numerik
9	dst_host_count	numerik
10	dst_host_srv_count	numerik
11	dst_host_same_src_port_rate	numerik
12	dst_host_serror_rate	numerik
13	dst_host_srv_serror_rate	numerik
14	flag	nominal

Sumber : (Song dkk., 2011)

2.4 K-means

K-means merupakan salah satu metode *clustering* yang sering dipakai proses klasifikasi data. Metode ini bertujuan untuk mempartisi data menjadi sejumlah K partisi dimana jarak antara data ke pusat partisi (*centroid*) minimal. Data-data tersebut dikelompokkan ke *cluster* dengan *centroid* yang memiliki jarak terdekat dari data tersebut.

Langkah-langkah dari metode k-means adalah sebagai berikut :

a. menentukan sejumlah K *centroid* $\{c_1, c_2, .., c_K\}$ secara acak. *Centroid* ini akan menjadi pusat masing-masing *cluster*,
b. mengelompokkan seluruh data x_i pada *dataset* ke *centroid* terdekat,
c. menentukan lokasi *centroid* baru dengan menghitung rata-rata data yang ada pada tiap *cluster*,
d. mengulangi langkah b dan c hingga lokasi *centroid* dari masing-masing *cluster* tidak berubah/bergeser.

Kelemahan dari metode ini adalah pada parameter K yang perlu ditentukan terlebih dahulu untuk mengetahui jumlah partisi yang diperlukan. Nilai K yang terlalu besar dapat mengakibatkan *overfit* atau data tidak terpartisi dengan baik (kurang homogen). Penentuan *centroid* yang dilakukan secara acak juga mempengaruhi kestabilan bentuk *cluster* yang dihasilkan.

2.5 K-medoid

Metode K-medoid memiliki kemiripan dengan metode K-means. Perbedaan utamanya terletak pada representasi pusat *cluster* yang digunakan.

Berbeda dengan k-means yang menentukan pusat *cluster* berdasarkan nilai-rata rata (*mean*), pusat *cluster* pada K-medoid adalah suatu data yang secara riil posisinya berada di tengah *cluster* (*median*). Hal ini menjadi keunggulan algoritma ini, karena posisi medoid tidak terlalu terpengaruh terhadap adanya *noise* ataupun distribusi jarak yang berbeda-beda antara data dengan pusat *cluster*.

Salah satu implementasi dari metode ini adalah algoritma PAM (*Partition Around Medoid*). Adapun langkah-langkah dari algoritma PAM (Kaufman dan Rousseeuw, 1990) adalah sebagai berikut :

a. menentukan sejumlah K data untuk inisialisasi *medoid* $\{o_1, o_2, .., o_K\}$ secara acak,
b. mengelompokkan seluruh data x_i pada *dataset* ke *medoid* terdekat,
c. menentukan lokasi *medoid* baru dengan cara mencari titik (data) pada *cluster* yang meminimalkan total jarak (*cost*) antara data dengan *medoid* baru.
d. mengulangi langkah b dan c hingga lokasi *medoid* dari masing-masing *cluster* tidak berubah/bergeser.

2.6 K-nearest neighbor

K-nearest neighbour (k-nn) adalah salah satu metode klasifikasi data yang umum digunakan karena sederhana dan cepat. Metode ini digunakan untuk menentukan kelas / label dari suatu data baru) berdasarkan jumlah tetangga terdekat yang dominan.

Parameter K pada metode k-nn digunakan untuk menentukan sejumlah k tetangga terdekat yang perlu dicari. Label data baru ditentukan berdasarkan *majority voting* label dari sejumlah k tetangga tersebut.

Kelemahan dari metode ini adalah pada parameter K yang harus ditentukan terlebih dahulu. Nilai K yang tidak sesuai dapat berakibat terjadinya kesalahan klasifikasi.

2.7 WEKA

WEKA (*Waikato Environment for Knowledge Analysis*) adalah sebuah perangkat lunak / *tools* untuk proses data mining yang dikembangkan oleh University of Waikato, Selandia Baru (Hall dkk., 2009). WEKA menyediakan

beragam algoritma *machine learning* yang dapat digunakan untuk melakukan analisis terhadap *dataset*. WEKA dapat digunakan untuk melakukan *data-processing*, klasifikasi, regresi, *clustering*, *association rules*, dan visualisasi.

Selain menyediakan antarmuka grafis (GUI), WEKA juga menyediakan *library* WEKA API yang dapat digunakan dalam pengembangan perangkat lunak untuk pemrosesan data terutama terkait pengembangan model *machine learning*. Library WEKA tersedia secara publik dan dapat dikembangkan sesuai kebutuhan implementasi dan pengujian.

BAB 3
METODOLOGI PENELITIAN

3.1 Rancangan Penelitian

Secara umum, penelitian ini diawali dengan studi literatur, perumusan masalah dan penyelesaiannya, perancangan sistem yang diajukan, implementasi sistem, serta diakhiri dengan uji coba dan analisis hasil. Sedangkan penyusunan laporan tesis dimulai pada awal sampai akhir penelitian. Secara lebih detail, penelitian ini dirancang dengan urutan seperti Gambar 3.1 dengan penjelasan sebagai berikut :

1. Studi literatur

   Mempelajari literatur berkaitan dengan dasar teori dan metode yang akan digunakan. Sumber literatur yang digunakan berupa literatur primer yang berasal dari jurnal ilmiah dan prosiding.

2. Perancangan Sistem

   Pada tahap ini dilakukan analisis yang meliputi perumusan masalah, batasan-batasan masalah, dan penyelesaiannya. Proses perancangan juga meliputi evaluasi keunggulan dan kelemahan dari metode yang sudah diajukan pada penelitian-penelitian sebelumnya. Berdasarkan evaluasi tersebut dilakukan perancangan metode yang akan diajukan.

3. Implementasi Sistem

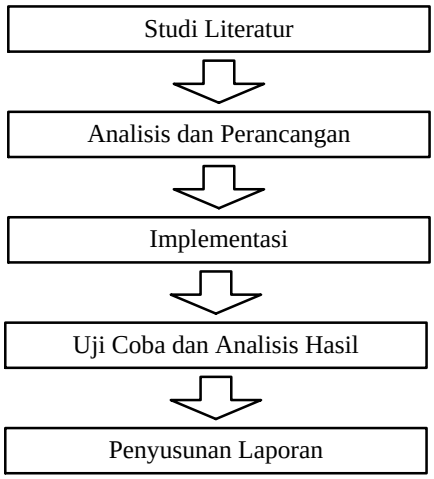
   Mengimplementasikan metode yang akan diajukan dengan menulis *pseudocode* dari metode tersebut sampai dihasilkan program yang siap untuk dieksekusi.

4. Uji coba dan Analisis Hasil

   Melakukan pengujian dan analisis terhadap hasil dan performa metode yang diajukan.

5. Penyusunan Laporan

   Setiap kegiatan yang dilakukan dalam penelitian ini didokumentasikan. Mulai dari tahap studi literatur sampai uji coba, semua ditulis dalam laporan tesis. Laporan tesis ditulis berdasarkan ketentuan yang berlaku.



Gambar 3.1 Tahapan Penelitian

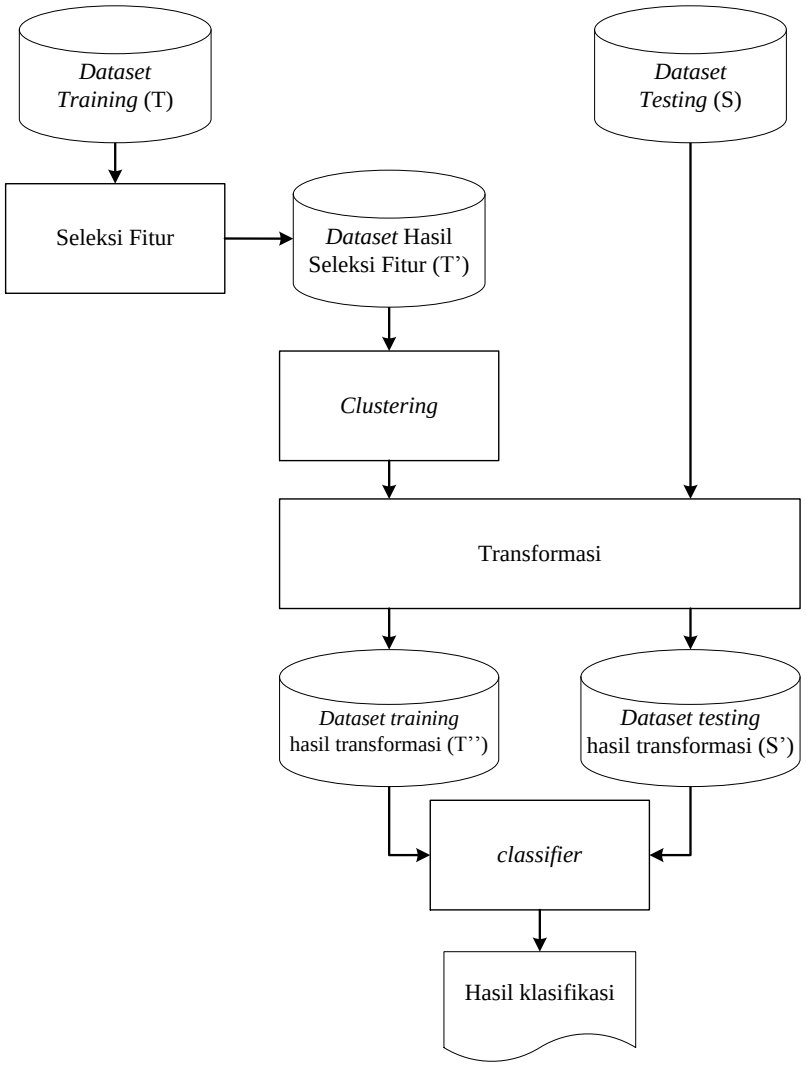
3.2 Rancangan Sistem

Secara umum, langkah-langkah pada metode yang diajukan serupa dengan tahapan pada pendeteksian intrusi dengan metode *centroid-based* lainnya seperti TANN (Tsai dan Lin, 2010), CANN (Lin dkk., 2015), dan metode (Muchammad dan Ahmad, 2015). Pendekatan *centroid-based* (Han dan Karpys, 2000) menyatakan bahwa jarak antara suatu data terhadap *centroid* suatu *cluster*, dapat memberikan informasi tambahan dalam menentukan tingkat kesamaan (*similarities*) label dari suatu data dengan *centroid*. Hal ini dikarenakan data-data yang memiliki tingkat kesamaan yang tinggi cenderung berada pada jarak yang berdekatan. Dengan menggunakan pendekatan ini, maka proses *training* cukup dilakukan berdasarkan posisi data terhadap data *centroid*, dengan demikian dapat menurunkan jumlah komputasi yang perlu dilakukan.

Muchammad dan Ahmad (2015) menentukan suatu *cluster* harus memenuhi tingkat homogenitas tertentu berdasarkan *threshold gini impurity index* dan entropi *cluster*. Jika ditemukan *cluster* yang yang melebihi *threshold* tersebut, maka harus dipartisi kembali menjadi 2 *cluster* yang lebih kecil.

Pada penelitian ini, penulis mengajukan penggunaan *threshold* ukuran *cluster* berdasarkan *radius cluster* tersebut. *Radius cluster* merupakan jarak dari *centroid* ke data terjauh dalam suatu *cluster*. Jika *radius cluster* melebihi nilai *threshold*, maka *cluster* harus dipartisi menjadi 2 *cluster* yang lebih kecil. Nilai

ambang radius dihitung berdasarkan jarak rata-rata antar data pada seluruh data *training*.



Gambar 3.2 Alur Sistem yang Diajukan

Sistem yang diajukan dalam penelitian ini merupakan sistem deteksi intrusi berbasis *machine learning* yang terdiri dari rangkaian proses seleksi fitur, *clustering*, transformasi fitur, dan klasifikasi sebagaimana diilustrasikan seperti pada Gambar 3.2.

Alur sistem yang diajukan terdiri dari 4 tahap sebagai berikut :

1. Sebelum proses *training* dimulai, dilakukan seleksi fitur terhadap *dataset training* (T) untuk menghilangkan fitur-fitur yang tidak signifikan sehingga dihasilkan *dataset* yang hanya memuat fitur yang relevan.

2. Pada proses *clustering*, *dataset* T' dikelompokkan menjadi beberapa *cluster* dengan ukuran *cluster* yang sesuai dengan berdasarkan nilai ambang *radius*. Selanjutnya dilakukan identifikasi *sub-medoid* dari masing-masing *cluster*.
3. Berdasarkan data *centroid* dan *sub-medoid* pada masing-masing *cluster*, dibangkitkan fitur satu dimensi yang merupakan transformasi dari seluruh fitur pada *dataset* T' sehingga dihasilkan *dataset training* baru yaitu T''. *Dataset* T'' digunakan untuk melakukan proses *training classifier*.
4. Data *testing* yang akan diklasifikasikan harus ditransformasikan ke bentuk satu dimensi dengan metode transformasi yang sama seperti pada penjelasan nomor 3. Hasil dari proses klasifikasi adalah label hasil deteksi apakah data tersebut merupakan data aktivitas normal atau serangan.

Penjelasan lebih detail dari masing-masing tahapan sistem deteksi intrusi diberikan pada bagian 3.2.1 sampai 3.2.5.

3.2.1. Ambang Radius (*Threshold* Radius)

Radius dari suatu cluster adalah jarak antara centroid dengan data terjauh pada suatu cluster. Nilai ini digunakan untuk menentukan ukuran dari suatu cluster. Pada *supervised learning* dan *semi-supervised learning* berlaku asumsi bahwa data-data yang memiliki label yang sama cenderung untuk saling berdekatan (Chapelle dkk., 2006). Kelompok data yang saling berdekatan tersebut dikelompokkan dalam suatu *cluster* dengan radius tertentu.

Han (2011) membahas tentang penggunaaan nilai jarak rata-rata antar data dalam proses *clustering* secara *hierarchical*. Nilai tersebut kemudian diolah untuk mendefinisikan suatu *high density area* yang merepresentasikan suatu *cluster*. Pada penelitian ini, diajukan suatu nilai yang disebut ambang radius ($r_{threshold}$) yang digunakan dalam proses *clustering* untuk membatasi ukuran *cluster* yang dihasilkan. Nilai $r_{threshold}$ dihitung sebagai jarak rata-rata antar data pada seluruh *dataset training* yang dihitung seperti persamaan (Han, 2011) sebagaimana yang terlihat pada persamaan 3.1 dan 3.2.

$$d_i = \frac{\sum_{j=1}^{n} d(X_i, X_j)}{n-1} \tag{3.1}$$

dengan :

d_i = adalah jarak rata-rata dari suatu data X_i ke data lainnya

i = $\{1, 2, \ldots, n\}, i \neq j$

n = jumlah data pada *dataset*

$d(X_i, X_j)$ = jarak (*euclidean distance*) antara data X_i dan X_j

$$r_{threshold} = \frac{\sum_{i=1}^{n} d_i}{n} \tag{3.2}$$

dengan :

$r_{threshold}$ = nilai ambang radius, jarak rata-rata antar data pada *dataset*

i = $\{1, 2, \ldots, n\}$

n = jumlah data pada *dataset*

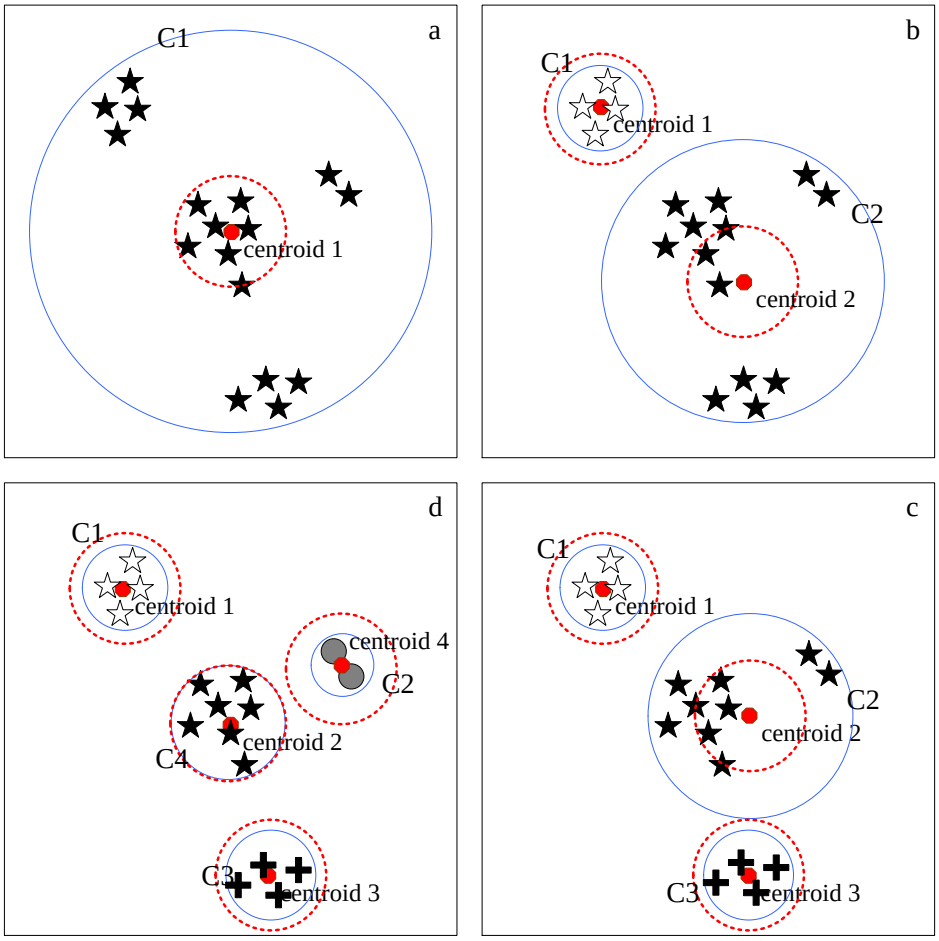
d_i = adalah jarak rata-rata dari suatu data X_i ke data yang lainnya

3.2.2. *Clustering*

Tahapan ini digunakan untuk memperoleh sejumlah nilai *centroid* yang akan digunakan dalam proses transformasi fitur. Tahapan ini menerapkan *divisive clustering* berbasis k-means dengan menggunakan nilai pada 3.2.1 sebagai *threshold* ($r_{threshold}$). Pada tahap awal proses, *dataset* dianggap sebagai suatu *cluster* besar. Jika *cluster* tersebut memiliki *radius* lebih besar dari nilai $r_{threshold}$ ($r_{cluster} > r_{threshold}$), maka *cluster* tersebut harus dipartisi lagi menjadi dua *cluster* yang lebih kecil. Pada awal *clustering* dipilih 2 data yang memiliki jarak terjauh sebagai *centroid* dari masing-masing *cluster*.

Ilustrasi contoh proses *clustering* ini dapat dilihat pada Gambar 3.4. Lingkaran putus-putus menyatakan $r_{threshold}$. Gambar 3.4.a adalah tahapan awal *clustering*. Pada tahap ini *cluster* C_1 dipartisi menjadi 2 bagian dengan metode k-means dengan menggunakan 2 data dengan jarak terjauh sebagai *centroid* dari masing-masing *cluster*. Hasil dari proses ini adalah *cluster* C_1 dan C_2 seperti pada Gambar 3.4.b. Setelah dievaluasi, cluster C_2 memiliki *radius* lebih besar dari $r_{threshold}$ (lingkaran putus-putus), sehingga *cluster* tersebut harus dipartisi menjadi 2 bagian. Proses ini dilakukan secara rekursif sampai semua *cluster* memiliki ukuran sesuai $r_{threshold}$. Pada Gambar 3.4.c dapat dilihat *cluster* C_2 telah dipartisi

menjadi *cluster* C_2 dan C_3. Kemudian pada Gambar 3.4.d *cluster* C_2 dipartisi kembali menjadi *cluster* C_2 dan C_4. Gambar 3.4.d mengilustrasikan kondisi akhir dari proses *clustering*, yaitu saat semua *cluster* memiliki ukuran yang memenuhi ambang $r_{threshold}$. $(r_{cluster} \leq r_{threshold})$.



Gambar 3.3 Ilustrasi Proses *Clustering*

3.2.3. Transformasi (Pembangkitan Fitur Baru)

Pada tahap trasformasi fitur dilakukan pembangkitan fitur baru yang merupakan penjumlahan jarak dari setiap data terhadap setiap *centroid* yang dihasilkan pada tahap sebelumnya.

Langkah-langkah untuk pembangkitan fitur baru mengadopsi langkah-langkah pada (Muchammad dan Ahmad, 2015) dengan menggunakan penjumlahan 2 macam jarak, yaitu : (i) jarak antara data dengan *centroid* tiap *cluster* dan (ii) jarak

antara data dengan *sub-centroid* dari *cluster* terdekat. Pada penelitian ini, jarak antara data dengan *sub-centroid* diganti dengan jarak antara data dengan *sub-medoid* dari *cluster* terdekatnya.

Sub-medoid untuk setiap *cluster* diperoleh dengan menerapkan algoritma PAM (*Partition Around Medoid*) ke masing-masing *cluster* yang dihasilkan pada tahap *clustering*. Jumlah *sub-medoid* yang diperlukan ditentukan dengan persamaan 3.3. Untuk *cluster* yang hanya terdiri dari 1 data, maka secara otomatis data tersebut merupakan *centroid* sekaligus *sub-medoid* dari *cluster* tersebut.

$$k = L \tag{3.3}$$

dengan :

k = jumlah *sub-medoid* yang akan dicari

L = jumlah kelas / label pada *cluster*

Setelah *sub-medoid* untuk tiap *cluster* diperoleh sesuai persamaan di atas, maka dibangkitkan fitur baru satu dimensi untuk seluruh *dataset training* dengan menggunakan persamaan tranformasi 3.4.

$$X_i' = \sum_{j=1}^{o} d(X_i, C_j) + \sum_{l=1}^{k} d(X_i, SM_l) \tag{3.4}$$

dengan :

X_i' = data hasil transformasi

i = $\{1, 2, \ldots, m\}$

m = jumlah data pada *dataset*

j = $\{1, 2, \ldots, o\}$

o = jumlah *centroid* pada *dataset*

X_i = data ke-i pada *dataset*

C_j = *centroid* ke-j pada *dataset*

l = $\{1, 2, \ldots, k\}$

k = jumlah *sub-medoid* pada *cluster* terkait

SM_l = *sub-medoid* ke-l pada *cluster* terkait

$d(X_i, C_j)$ = jarak (*euclidean distance*) antara data X_i dengan *centroid* C_j

$d(X_i, SM_l)$ = jarak (*euclidean distance*) antara data X_i dan *sub-medoid* SM_l

3.2.4. Klasifikasi

Pada tahap klasifikasi, setiap data *testing* terlebih dahulu ditransformasikan dengan proses yang sama seperti pada 3.2.3. Langkah pertama adalah mengidentifikasi *cluster* terdekat dari data tersebut. Langkah berikutnya adalah melakukan penghitungan jarak antara data tersebut dengan seluruh *centroid* dan melakukan penghitungan jarak antara data tersebut dengan seluruh *sub-medoid* pada *cluster* terdekat. Hasil transformasi data *testing* tersebut adalah penjumlahan dari jarak antara data dengan *centroid* dan jarak antara data dengan seluruh *sub-medoid* pada *cluster* terdekat yang sesuai dengan persamaan 3.3 dan 3.4.

Classifier yang digunakan adalah algoritma k-nn. *Classifier* ini dilatih terlebih dahulu dengan data traning yang sudah ditransformasikan ke satu dimensi pada bagian 3.2.4. Hasil dari proses klasifikasi ini adalah kelas / label untuk data *testing* tersebut.

Keseluruhan proses *clustering* sampai dengan *klasifikasi* dapat diamati melalui *pseudocode* pada gambar 3.4 dan 3.5 berikut.

No	*Pseudocode* div_kmeans : *Input* : D[], dataset $r_{threshold}$, ambang radius cluster *Output* : C[], *array of clusters*
1	/* *divisive k-means (recursive)* */
2	C[] = {}
3	C_{temp}[] = kmeans(D, 2)
4	foreach C_i in C_{temp}[]
5	if ($r_i \geq r_{threshold}$)
6	C[] = {C[], divisive_kmeans(C_i, 2, r_{max})}
7	else
8	C[] = {C[], C_i}
9	return C[]

Gambar 3.4 *Pseudocode* Divisive K-Means

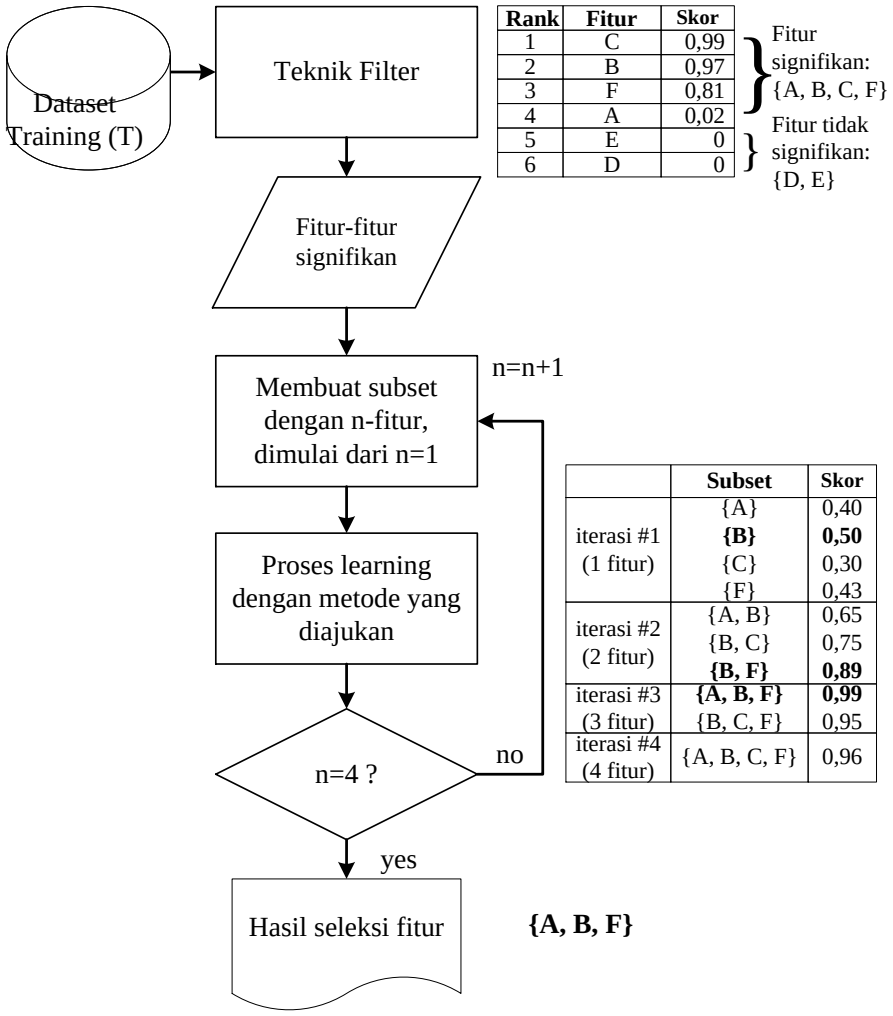
No	*Pseudocode* proposed_method : *Input* : D_train[], *dataset* untuk *training* D_test[], *dataset* untuk *testing*
1	/* *CLUSTERING* */
2	T = D_train[]
3	/* *Menghitung nilai dari* $r_{threshold}$ */
4	foreach D_i in D_train[]
5	$d_i = \frac{\sum_{j=1}^{n} d(D_i, D_j)}{n-1}$
6	$r_{\text{threshold}} = \frac{\sum_{i=1}^{n} d_i}{n}$
7	/* *Mengelompokkan dataset ke dalam beberapa cluster dengan ambang batas radius* */
8	C[] = divisive_kmeans(T, $r_{\text{threshold}}$)
9	/* *Mengidentifikasi sub-medoid – sub-medoid dari setiap cluster yang terbentuk* */
10	foreach C_i in C[]
11	SM[] = kmedoid(C_i, num_sub_medoid)
12	/* *Transformasi fitur ke 1 dimensi* */
13	T' = {}
14	foreach D_t in T
15	c = get_nearest_cluster(D_t)
16	$D'_t = \sum_{j=1}^{o} d(D_t, C_j) + \sum_{l=1}^{k} d(D_t, SMc_l)$
17	T' = {T', D'_t}
18	S = D_test[]
19	S' = {}
20	foreach D_s in S
21	c = get_nearest_cluster(D_s)
22	$D'_s = \sum_{j=1}^{o} d(D_s, C_j) + \sum_{l=1}^{k} d(D_s, SMc_l)$
23	S' = {S', D'_s}
24	
25	/* *CLASSIFICATION* */
26	foreach D'_s in S'
27	c = get_cluster(D'_s)
28	$D'_{t_c} = get\_member(c)$
29	$class_s$ = knn(D'_{t_c}, D'_s, k)

Gambar 3.5 *Pseudocode* Metode Yang Diajukan

3.2.5. Seleksi Fitur

Tahapan ini diperlukan untuk menghilangkan fitur-fitur yang tidak signifikan dari *dataset* yang dapat mempengaruhi akurasi dari metode yang diajukan. Seleksi fitur akan dilakukan dengan menggunakan penggabungan dari teknik *filter* dan teknik *wrapper*. Metode ini diawali dengan seleksi fitur dengan menggunakan teknik *filter* untuk memperoleh daftar fitur yang secara statistik memiliki pengaruh dalam penentuan kelas pada *dataset*. Langkah berikutnya adalah membentuk *subset-subset* fitur dari daftar fitur tersebut untuk dievaluasi dengan

menggunakan model sistem deteksi intrusi yang diajukan sehingga diperoleh *subset* fitur yang memberikan hasil terbaik.



Gambar 3.6 Contoh Alur Proses Seleksi Fitur untuk *Dataset* dengan 6 Fitur

Sebagai contoh, Gambar 3.3 mengilustrasikan contoh proses seleksi fitur untuk *dataset* yang memiliki 6 fitur, yaitu {A, B, C, D, E, F}. Proses dimulai dengan melakukan evaluasi fitur-fitur tersebut dengan menggunakan teknik *filter*. Dari proses ini dihasilkan 2 kelompok fitur, yaitu kelompok fitur signifikan {A, B, C, F} dan kelompok fitur yang tidak signifikan {D, E}. Suatu fitur disebut tidak signifikan jika hasil evaluasi pengaruhnya dalam penentuan kelas data bernilai 0. Fitur tidak signifikan ini tidak perlu disertakan dalam proses *learning.*

Langkah berikutnya adalah melakukan pemilihan fitur secara bertahap (iterasi n-fitur) dengan mengevaluasi performansi masing-masing *subset* yang dapat dibentuk dari kelompok fitur signifikan tersebut dengan menggunakan model sistem deteksi intrusi yang diajukan. Proses pemilihan fitur tersebut menggunakan *forward selection hill climbing search* seperti pada (Hall dan Holmes, 2003). Seperti pada Gambar 3.3, pada iterasi pertama (1 fitur), dilakukan pengujian perfoma dari masing-masing *subset* terhadap model. Dari hasil iterasi pertama, diketahui ternyata *subset* {B} memiliki performansi terbaik daripada *subset* 1 fitur lainnya. Untuk itu pada iterasi selanjutnya (iterasi kedua), *subset* yang perlu dievaluasi adalah *subset* yang memuat fitur B saja, yaitu *subset* {A, B}, {B, C}, dan {B, F}. Dari hasil iterasi kedua, diketahui *subset* {B, F} mengungguli *subset* 2 fitur lainnya. Proses ini dilakukan sampai iterasi terakhir, yaitu iterasi ke 4. Berdasarkan hasil evaluasi, kemudian diketahui bahwa *subset* yang memberikan hasil terbaik adalah *subset* 3 fitur, yaitu {A. B, F}. Dengan demikian fitur {A, B, F} dipilih sebagai fitur yang akan digunakan dalam proses selanjutnya.

3.3 Rancangan Pengujian

Setelah tahapan perancangan sistem, penelitian dilanjutkan dengan melakukan uji coba terhadap sistem yang telah dibuat untuk memperoleh hasil pengujian dan dilakukan analisis/evaluasi hasil tersebut. Tahap uji coba dilaksanakan untuk mengetahui apakah penelitian yang dilakukan telah memenuhi tujuan penelitian. Sebagaimana disebutkan di atas, tujuan dari penelitian ini adalah untuk melakukan seleksi fitur pada *dataset* dan mendapatkan model sistem deteksi intrusi yang lebih baik dari metode sebelumnya.

Untuk melakukan pengujian, dilakukan implementasi metode yang diajukan dan dua metode pembanding dengan menggunakan bahasa pemrograman Java dan *library* yang tersedia pada WEKA versi 3.6. Metode yang digunakan sebagai pembanding adalah metode CANN dan metode (Muchammad dan Ahmad, 2015). Selanjutnya dalam penelitian ini, metode CANN disebut sebagai metode B1 (metode pembanding 1), metode Muchammad dan Ahmad (2015) disebut metode B2 (metode pembanding 2) dan metode yang diajukan disebut metode P. Pada penelitian ini, juga diimplementasikan algoritma PAM yang digunakan untuk

memperoleh nilai sub-medoid dari *cluster* yang dihasilkan, karena implementasi algoritma tersebut belum tersedia pada library WEKA yang digunakan. Untuk implementasi algoritma yang sudah tersedia pada WEKA, seperti implementasi k-nn dan kmeans, tidak dilakukan implementasi ulang. Proses klasifikasi pada masing-masing metode (B1, B2, dan P) dilakukan dengan algoritma k-nn dengan menggunakan nilai k = 3 seperti pada penelitian (Muchammad dan Ahmad, 2015).

Pada proses pengujian, digunakan teknik validasi *10-fold cross validation* (10-fold CV). Teknik validasi ini dilakukan dengan cara membagi *dataset* menjadi 10 bagian. Kemudian pada setiap kali pengujian, digunakan kombinasi dari 9 bagian untuk proses *training* dan 1 bagian lainnya untuk proses *testing*. Dengan demikian jumlah pengujian yang dilakukan adalah sebanyak 10 kali terhadap *dataset* untuk setiap metode yang diuji.

3.3.1. *Dataset*

Dataset yang digunakan dalam penelitian ini adalah *dataset* NSL-KDD dan Kyoto2006+. Kedua *dataset* ini tersedia untuk publik dan banyak digunakan oleh para peneliti pada topik sistem deteksi intrusi pada jaringan.

Implementasi metode dengan menggunakan *library* WEKA mempersyaratkan *dataset* masukan berupa format file ARFF atau CSV, untuk itu dilakukan transformasi terlebih dahulu untuk *dataset* yang belum tersedia dalam format file ARFF.

Dataset NSL-KDD yang digunakan dalam penelitian ini adalah *dataset* KDD-Train+_20Percent.csv yang diperoleh dari laman (https://web.archive.org/web/20150604025119/http://nsl.cs.unb.ca/NSL-KDD/). *Dataset* ini kemudian ditransformasi ke format ARFF yang dapat dikenali oleh library WEKA. Proses selanjutnya adalah menghilangkan fitur level yang tidak digunakan dalam eksperimen ini sehingga dihasilkan dataset yang terdiri dari 41 fitur seperti pada bagian 2.2 dan 1 fitur *class*. *Dataset* ini kemudian dikonversi ke bentuk *5-class problem* sesuai dengan kelompok serangan seperti pada Tabel 3.1. Proses konversi tersebut dilakukan sesuai dengan pengelompokan *attack* pada KDD-99 (Hettich dan Bay, 1999) seperti pada kajian pustaka bagian 2.2 dengan tujuan supaya jumlah kelas / label yang digunakan sesuai dengan hasil penelitian (Lin dkk., 2015) dan (Muchammad dan Ahmad, 2015). Selanjutnya *dataset*

NSLKDD tersebut digunakan untuk membentuk 4 *dataset* lainnya dengan komposisi seperti pada Tabel 3.2 dengan cara menghilangkan fitur-fitur yang tidak diperlukan dan menghilangkan data-data redundan yang muncul akibat proses tersebut.

Dataset Kyoto yang digunakan dalam penelitian ini adalah *dataset* 20090730.txt yang merupakan rekaman akses ke *honeypot* pada tanggal 30 Juli 2009 (diperoleh dari laman http://www.takakura.com/Kyoto_data). Kemudian *dataset* ini ditransformasikan ke format ARFF. Fitur yang digunakan adalah fitur no 1 s.d. 14 seperti pada Tabel 2.3 dan 1 fitur *label*, sedangkan fitur sisanya tidak digunakan. *Dataset* ini kemudian dikonversi ke bentuk *2-class problem* seperti pada Tabel 3.1. Proses ini dilakukan supaya jumlah kelas / label yang digunakan pada penelitian ini sesuai dengan yang digunakan pada penelitian (Lin dkk., 2015) dan (Muchammad dan Ahmad, 2015). Seperti halnya tahapan preproses pada NSLKDD, selanjutnya *dataset* Kyoto tersebut digunakan untuk membentuk 2 *dataset* lainnya dengan komposisi seperti Tabel 3.2 dengan cara menghilangkan fitur-fitur yang tidak diperlukan dan menghilangkan data-data redundan yang muncul akibat proses tersebut. Proses penghilangan data yang redundan ini bertujuan untuk meminimalkan bias yang terjadi saat proses *learning* (Tavallaee dkk., 2009).

Tabel 3.1 Peta Konversi Kelas pada *Dataset* NSLKDD dan Kyoto

No	Nama *Dataset*	Kelas / Label Sebelum Konversi	Kelas / Label Hasil Konversi
1.	NSLKDD	Normal	normal
		back, land, neptune, pod, smurf, teardrop	dos
		buffer_overflow, loadmodule, perl, rootkit	u2r
		ipsweep, nmap, portsweep, satan	probe
		ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster	r2l
2.	Kyoto	1	normal
		-1, -2	attack

Tabel 3.2 Daftar *Dataset* yang Digunakan dalam Eksperimen

No	Nama *Dataset*	Fitur	Komposisi Data
1.	NSLKDD-6	land, urgent, num_failed_logins, num_shells, num_outbound_cmds, is_host_login, class	Normal = 6 DoS = 2 R2L = 3 Probe = 2 U2R = 2 Total = 15
2.	NSLKDD-8	src_bytes, dst_bytes, wrong_fragment, hot, count, diff_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, class	Normal = 12588 DoS = 3910 R2L = 110 Probe = 966 U2R = 11 Total = 17585
3.	NSLKDD-19	protocol_type, src_byte, flag, logged_in, count, serror_rate, rerror_rate, same_srv_rate, srv_serror_rate, srv_rerror_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_srcport_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, class	Normal = 13206 DoS = 8690 R2L = 209 Probe = 1710 U2R = 11 Total = 23826
4.	NSLKDD-P	Diperoleh dari hasil seleksi fitur.	Diperoleh dari hasil seleksi fitur
5.	Kyoto-7	duration, source_bytes, destination_bytes, count, same_srv_rate, srv_serror_rate, dst_host_srv_count, label	Normal = 59560 Attack = 4453 Total = 64013
6.	Kyoto-P	Diperoleh dari hasil seleksi fitur.	Diperoleh dari hasil seleksi fitur.

Tabel 3.2 memuat daftar *dataset* yang digunakan dalam penelitian ini. NSLKDD-6, NSLKDD-8, NSLKDD-19, dan NSLKDD-P adalah *dataset* NSLKDD yang telah dipreproses seperti pada tahapan-tahapan di atas. NSLKDD-6 dan NSLKDD-19 merupakan *dataset* NSLKDD yang telah dipreproses dan memiliki 6 fitur dan 19 fitur seperti yang digunakan pada (Lin dkk., 2015). NSLKDD-8 adalah *dataset* NSLKDD yang telah dipreproses dan memiliki 8 fitur

seperti pada (Muchammad dan Ahmad, 2015). *Dataset* Kyoto-7 adalah *dataset* Kyoto yang telah dipreproses dan memuat 7 fitur seperti pada (Muchammad dan Ahmad, 2015). Sedangkan NSLKDD-P dan Kyoto-P masing-masing adalah *dataset* NSLKDD dan Kyoto yang akan dipreproses berdasarkan hasil eksperimen seleksi fitur pada penelitian ini.

3.3.2. Eksperimen Seleksi Fitur

Pada eksperimen ini dilakukan pengujian untuk mengetahui *subset fitur* dari NSLKDD dan Kyoto yang merupakan fitur paling signifikan. Tahapan pertama dari eksperimen ini adalah melakukan seleksi fitur dengan teknik *filter* untuk mengeliminasi fitur-fitur yang tidak signifikan dengan menggunakan evaluator *Gain Ratio*. Fitur-fitur dari dataset yang memiliki nilai *Gain Ratio* sama dengan nol akan dieliminasi. Tahapan ini dilakukan dengan menggunakan fitur *Select attributes* yang tersedia pada aplikasi WEKA.

Tahapan selanjutnya adalah melakukan pengujian subset fitur signifikan yang dihasilkan dari tahapan pertama di atas. Pada proses pengujian ini, dilakukan preproses terhadap *dataset* dengan menghilangkan fitur-fitur yang tidak digunakan dan menghilangkan data yang redundan seperti pada bagian 3.2.5. Pada setiap akhir pengujian suatu subset fitur, akan dihitung nilai *sensitivity* dan *specificity* dari subset fitur tersebut. Pada penelitian ini, *sensitivity* didefinisikan sebagai tingkat kemampuan dari metode deteksi dalam mengenali setiap aktifitas serangan, sedangkan *specificity* didefinisikan sebagai tingkat kemampuan dari metode deteksi dalam mengenali setiap aktifitas normal. Penjelasan lebih detil tentang *specificity* dan *sensitivity* dapat diperoleh pada bagian 3.3.5.

Nilai *sensitivity* dan *specificity* yang tinggi dapat membentuk *accuracy* yang tinggi. Pada penelitian ini, subset fitur yang dipilih sebagai subset fitur terbaik dari *dataset* adalah subset fitur yang memiliki nilai *sensitivity* dan *specificity* yang berimbang yang dapat meningkatkan *accuracy*. Pada setiap iterasi akan dilakukan pembandingan nilai dengan menggunakan persamaan 3.5, 3.6, dan 3.7. Jika ditemukan 2 subset dengan *score* sama, maka subset yang memiliki fitur dengan nilai Gain Ratio tertinggi dipilih sebagai subset terbaik pada iterasi tersebut. Subset terbaik dari setiap iterasi kemudian akan dibandingkan dengan cara yang sama seperti di atas sehingga diperoleh subset terbaik untuk keseluruhan iterasi.

Pseudocode dari proses pembandingan nilai / *score* setiap subset dapat dilihat pada Gambar 3.7.

$$\bar{A}_i = \frac{Sens_i + Spec_i}{2} \tag{3.5}$$

$$|A|_i = |Sens_i - Spec_i| \tag{3.6}$$

$$Score_i = \bar{A}_i - |A|_i \tag{3.7}$$

dengan :

$Score_i$ = nilai hasil evaluasi subset fitur ke-i

$\bar{A}_i$ = rata-rata nilai *sensitivity* dan *specificity* subset fitur ke-i

$|A|_i$ = selisih nilai *sensitivity* dan *specificity* subset fitur ke-i

$Sens_i$ = nilai *sensitivity* subset fitur ke-i

$Spec_i$ = nilai *specificity* subset fitur ke-i

No	*Pseudocode* compare_subset : *Input* : feature$_1$, subset fitur 1 feature$_2$, subset fitur 2 *Output* : false, jika subset 1 tidak lebih baik dari subset 2 true, jika subset 1 lebih baik dari subset 2
1	*/* membandingkan apakah subset 1 lebih baik dari subset 2 */*
2	foreach i in {1, 2}
3	$\bar{A}_i = (sens_i + spec_i)/2$ // rata-rata sensitivity dan specificity
4	$\|A\|_i = \|sens_i - spec_i\|$ // selisih sensitivity dan specificity
5	$Score_i = \bar{A}_i - \|A\|_i$
6	if ($Score_1 < Score_2$)
7	return false
8	else if ($Score_1 == Score_2$)
9	if (length(feature$_1$) < length(feature$_2$))
10	// jika jumlah fitur feature$_1$ lebih sedikit dari jumlah fitur feature$_2$
11	return true
12	else if (length(feature$_1$) == length(feature$_2$))
13	rank$_1$ = get_attribute_rank(feature$_1$)
14	rank$_2$ = get_attribute_rank(feature$_2$)
15	if (rank1 < rank2)
16	// jika nilai gain ratio feature$_1$ lebih kecil dari nilai gain ratio feature$_2$
17	return false
18	else
19	return true
20	else
21	return false
22	else
23	return true

Gambar 3.7 *Pseudocode* Fungsi Pembandingan Subset

Seleksi fitur dilakukan dengan menggunakan metode P pada *dataset* NSLKDD dan Kyoto yang telah dipreproses dan menggunakan nilai k=3 pada proses klasifikasi. Pada proses *clustering*, digunakan 2 data yang memiliki jarak euclidean terjauh sebagai *centroid* awal (*initial centroid*). Dengan cara ini diharapkan diperoleh jumlah *cluster* terbaik yang dapat mewakili kondisi data. Subset fitur terbaik dari masing-masing seleksi fitur pada *dataset* NSLKDD dan Kyoto digunakan untuk membentuk *dataset* NSLKDD-P dan *dataset* Kyoto-P seperti pada Tabel 3.2.

3.3.3. Eksperimen metode pembanding B1 dan B2 dan metode yang diusulkan P

Pada proses ini dilakukan pengujian masing-masing metode terhadap *dataset* pada Tabel 3.2. Secara umum, akan dilakukan sebanyak 18 pengujian seperti pada Tabel 3.3 dengan menggunakan teknik validasi *10-fold cross validation*. Hal yang ingin diketahui dari eksperimen ini adalah : (a) apakah metode yang diusulkan dapat memberikan hasil lebih baik dari metode pembanding untuk setiap *dataset* pengujian dan (b) apakah penggunaan *dataset* hasil seleksi fitur dapat digunakan untuk meningkatkan performansi dari masing-masing metode.

Tabel 3.3 Daftar Eksperimen Metode Pembanding dan Metode yang Diusulkan terhadap *Dataset*

No	Nama *Dataset*	Nama Metode
1.	NSLKDD-6	Metode B1
2.	NSLKDD-6	Metode B2
3.	NSLKDD-6	Metode P
4.	NSLKDD-8	Metode B1
5.	NSLKDD-8	Metode B2
6.	NSLKDD-8	Metode P
7.	NSLKDD-19	Metode B1
8.	NSLKDD-19	Metode B2
9.	NSLKDD-19	Metode P
10.	NSLKDD-P	Metode B1
11.	NSLKDD-P	Metode B2
12.	NSLKDD-P	Metode P
13.	Kyoto-7	Metode B1

No	Nama *Dataset*	Nama Metode
14.	Kyoto-7	Metode B2
15.	Kyoto-7	Metode P
16.	Kyoto-P	Metode B1
17.	Kyoto-P	Metode B2
18.	Kyoto-P	Metode P

Pengujian masing-masing metode menggunakan nilai k=3 pada proses klasifikasi dengan k-nn. Untuk proses *clustering* dengan k-means pada metode B1, digunakan nilai k=5 pada *dataset* NSLKDD-6, NSLKDD-8, NSLKDD-19, NSLKDD-P dan nilai k=2 pada *dataset* Kyoto-7, dan Kyoto-P. Pengujian dengan menggunakan metode B2 dilakukan dengan menggunakan parameter U (*threshold gini impurity index*) dan O (jumlah maksimum *sub-centroid* yang dievaluasi dalam suatu *cluster*) yang memberikan hasil terbaik sesuai hasil penelitian (Muchammad dan Ahmad, 2015) seperti pada Tabel 3.4. Pada setiap akhir eksperimen, dilakukan penghitungan nilai *sensitivity*, *specificity*, *accuracy*, dan *F-measure*.

Tabel 3.4 Nilai U dan O yang digunakan pada metode pembanding-2 untuk masing-masing *dataset*

No	Nama *Dataset*	U	O
1.	NSL-KDD	0,2	4
2.	Kyoto-2006+	0,1	6

Sumber : (Muchammad dan Ahmad, 2015)

3.3.4. Eksperimen Pembandingan Penghitungan Jarak dari Metode pembanding B2 dan Metode yang diusulkan P pada NSLKDD-P

Eksperimen ini dilakukan untuk mengetahui apakah : (i) pembatasan ukuran *cluster* dengan ambang *radius* dapat digunakan untuk menggantikan pembatasan dengan ambang *gini impurity index* seperti pada metode B2 dan (ii) apakah penggunaan jarak ke *sub-medoid* dapat digunakan untuk menggantikan penggunaan jarak ke *sub-centroid*. Secara umum, akan dilakukan sebanyak 6 pengujian terhadap dataset NSLKDD-P dengan skenario seperti pada Tabel 3.5 dengan menggunakan teknik validasi *10-fold cross validation*.

Tabel 3.5 Skenario Eksperimen Pembandingan Penghitungan Jarak dari Metode B2 dan Metode P pada NSLKDD-P

No	Metode	*Threshold* yang Digunakan	Hitung Jarak ke Subcentroid / Submedoid
1.	B2	*gini impurity index*	tidak dihitung
2.	B2	*gini impurity index*	*sub-centroid*
3.	B2	*gini impurity index*	*sub-medoid*
4.	P	*radius*	tidak dihitung
5.	P	*radius*	*sub-centroid*
6.	P	*radius*	*sub-medoid*

3.3.5. Analisis Hasil

Pada tahapan ini dilakukan analisa terhadap hasil yang telah dicatat pada tahap pengujian. Dari studi literatur, diketahui terdapat beberapa metrik evaluasi yang dapat digunakan untuk analisa kemampuan model deteksi intrusi, namun untuk penelitian ini digunakan : *sensitivity*, *specificity*, *accuracy*, dan *F-measure*.

Sensitivity, *specificity*, dan *accuracy* digunakan untuk memperoleh tingkat kebenaran klasifikasi. *Sensitivity* didefinisikan sebagai tingkat kemampuan dari metode deteksi untuk mengenali seluruh data serangan dengan benar, sedangkan *specificity* didefinisikan sebagai tingkat kemampuan dari metode deteksi dalam mengenali seluruh data normal dengan benar. *Accuracy* menyatakan kemampuan IDS dalam melakukan deteksi aktivitas serangan dan normal dengan benar.

Dalam proses deteksi, sebuah model dapat memiliki *sensitivity* (disebut juga sebagai *recall*) yang sangat tinggi sehingga mampu mendeteksi sebagian besar serangan namun disertai terjadinya misklasifikasi data sehingga sebagian aktifitas normal juga terdeteksi sebagai serangan. Kondisi ini dapat dievaluasi dengan menggunakan nilai *precision* yang menyatakan persentase data aktifitas serangan yang benar pada keseluruhan hasil deteksi serangan. Untuk itu sebuah model deteksi intrusi yang baik juga harus memiliki nilai *sensitivity* yang tinggi sekaligus memiliki nilai *precision* yang tinggi pula.

F-measure merupakan ukuran akurasi berdasarkan rata-rata harmonik (*harmonic mean*) antara nilai *recall* (*sensitivity*) dan *precision* dari suatu metode deteksi intrusi. Semakin tinggi nilai *F-measure*, maka semakin baik model deteksi

intrusi dalam mendeteksi aktifitas serangan dan meminimalkan terjadinya misklasifikasi aktivitas normal sebagai serangan. *F-measure* digunakan dalam penelitian ini untuk mengkonfirmasi hasil evaluasi dari ketiga metrik evaluasi lainnya. Metrik-metrik tersebut dihitung dengan menggunakan persamaan 3.8, 3.9, 3.10, dan 3.11.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.8}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.9}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3.10}$$

$$F - measure = \frac{2 * TP}{2 * TP + FN + FP} \tag{3.11}$$

Nilai TP, TN, FP, dan FN diperoleh dengan menggunakan *confusion matrix* seperti pada Tabel 3.6. TP menyatakan jumlah serangan yang berhasil dideteksi sebagai serangan, TN menyatakan jumlah aktivitas normal yang berhasil dideteksi sebagai aktivitas normal, FP menyatakan jumlah aktivitas normal yang dideteksi sebagai serangan, dan FN menyatakan jumlah serangan yang tidak berhasil dideteksi.

Tabel 3.6 *Confusion Matrix* dengan 2 kelas

	Prediksi (hasil deteksi)	
Aktual	Normal	Serangan
Normal	TN	FP
Serangan	FN	TP

BAB 4
HASIL DAN PEMBAHASAN

4.1 Hasil Seleksi Fitur

Proses seleksi fitur dilakukan dengan 10-*folds cross validation*. Hasil dari proses ini adalah daftar fitur signifikan dari masing-masing *dataset* NSLKDD dan Kyoto. Daftar fitur signifikan ini kemudian digunakan untuk membentuk *dataset* NSLKDD-P dan Kyoto-P sebagaimana dimuat dalam Tabel 4.1.

Seleksi fitur tahap awal dilakukan dengan teknik filter menghitung Gain Ratio dari masing-masing fitur pada *dataset* dengan menggunakan *Select attributes* yang ada pada kakas bantu *WEKA Explorer*. *Attribute evaluator* yang digunakan adalah GainRatioAttributeEval dan *Search method* yang digunakan adalah Ranker. Parameter yang digunakan adalah parameter *default* yang disediakan pada kakas bantu tersebut.

Hasil dari seleksi fitur tahap pertama adalah fitur-fitur signifikan seperti pada Tabel 4.1. Dari hasil evaluasi tersebut diperoleh bahwa fitur yang tidak signifikan dari *dataset* NSLKDD adalah *num_outbounds_cmd* dan *is_host_login* yang memiliki nilai Gain Ratio = 0. Sedangkan untuk *dataset* Kyoto semua fitur merupakan fitur signifikan. Fitur-fitur signifikan dari hasil evaluasi ini, kemudian akan dievaluasi kembali melalui proses seleksi fitur tahap kedua dengan teknik *wrapper* seperti yang dibahas pada bagian 3.2.1.

Hasil dari evaluasi dari seleksi fitur tahap kedua adalah seperti Tabel 4.2. Dari 41 fitur pada NSLKDD (tidak termasuk fitur *class*) dapat direduksi menjadi 19 fitur, sedangkan untuk *dataset* Kyoto, hasil optimal dapat diperoleh dengan menggunakan seluruh fitur. *Dataset* hasil seleksi fitur ini akan digunakan dalam eksperimen selanjutnya dan untuk membedakan dengan *dataset* lainnya masing-masing *dataset* tersebut disebut sebagai *dataset* NSLKDD-P dan Kyoto-P.

Subset yang memiliki performansi terbaik untuk tiap-tiap n subset pada masing-masing *dataset* dapat dilihat pada Tabel 4.3 dan Tabel 4.4. Pada Tabel 4.3 dapat dilihat bahwa subset fitur nomor 24 {duration, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell,

su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate} memiliki nilai *accuracy* tertinggi yaitu sebesar 97,66%. Namun yang dipilih sebagai subset terbaik adalah subset fitur nomor 19 {duration, service, flag, land, wrong_fragment, urgent, num_failed_logins, root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate} karena memiliki pasangan nilai *sensitivity* dan *specificity* yang terbaik (berimbang) daripada subset fitur lainnya yang ditandai dengan *score* hasil evaluasi tertinggi berdasarkan persamaan 3.5, 3.6, dan 3.7. Demikian juga dengan Tabel 4.4, walaupun subset fitur nomor 12 {duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag} memiliki nilai accuracy tertinggi dibandingkan subset lainnya, namun yang dipilih sebagai subset fitur terbaik adalah subset nomor 14 {duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_count, dst_host_srv_count, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag} karena berdasarkan hasil evaluasi persamaan 3.7 memiliki nilai pemilihan subset fitur tertinggi, yang artinya memiliki rata-rata *sensitivity* dan *specificity* dan selisih keduanya yang terbaik dibandingkan subset lainnya.

Tabel 4.1 Hasil Seleksi Fitur Tahap Pertama

No	Nama *Dataset*	Fitur Signifikan	Fitur Tidak Signifikan
1.	NSLKDD-P	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, 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	num_outbound_cmds, is_host_login
2.	Kyoto-P	duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_count, dst_host_srv_count, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag	tidak ada

Tabel 4.2 Daftar *Dataset* yang Dibentuk Berdasarkan Hasil Seleksi Fitur Tahap Kedua

No	Nama *Dataset*	Fitur	Jumlah data
1.	NSLKDD-P	duration, service, flag, land, wrong_fragment, urgent, num_failed_logins,root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, class	Normal = 7016 DoS = 8939 R2L = 70 Probe = 1298 U2R = 11 Total = 17334
2.	Kyoto-P	duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_count, dst_host_srv_count, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag, label	Normal = 63263 Attack = 5758 Total = 69021

Tabel 4.3 Hasil Evaluasi Performansi Subset Fitur pada NSLKDD

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
1	{duration}	79,18	62,81	88,20	50,12
2	{duration, flag}	88,19	81,27	92,22	75,80
3	{duration, flag, is_guest_login}	89,09	83,96	91,99	79,95
4	{duration, service, flag, is_guest_login}	87,46	84,32	89,88	81,54
5	{duration, service, flag, is_guest_login, srv_count}	89,17	89,41	88,71	88,36
6	{duration, service, flag, is_guest_login, count, srv_count}	96,23	96,63	95,01	94,20
7	{duration, service, flag, is_guest_login, count, srv_count, dst_host_same_src_port_rate}	96,63	96,60	96,69	96,56
8	{duration, service, flag, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	97,07	97,01	97,17	96,93
9	{duration, service, flag, land, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	97,06	97,01	97,14	96,95
10	{duration, service, flag, land, urgent, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	97,06	97,01	97,15	96,94

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
11	{duration, service, flag, land, urgent, num_shells, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	96,86	96,78	96,99	96,68
12	{duration, service, flag, land, wrong_fragment, urgent, num_shells, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	97,05	96,99	97,15	96,91
13	{duration, service, flag, land, wrong_fragment, urgent, su_attempted, num_shells, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	96,96	96,98	96,94	96,92
14	{duration, service, flag, land, wrong_fragment, urgent, su_attempted, num_file_creations, num_shells, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_same_src_port_rate}	96,92	96,95	96,88	96,85
15	{duration, service, flag, land, wrong_fragment, urgent, su_attempted, num_file_creations, num_shells, is_guest_login, count, srv_count, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	97,34	97,21	97,54	97,05
16	{duration, service, flag, land, wrong_fragment, urgent, su_attempted, num_file_creations, num_shells, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	97,40	97,37	97,45	97,33
17	{duration, service, flag, land, wrong_fragment, urgent, num_failed_logins, su_attempted, num_file_creations, num_shells, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	97,36	97,34	97,39	97,32
18	{duration, service, flag, land, wrong_fragment, urgent, num_failed_logins, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	97,45	97,30	97,67	97,12

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
19	{duration, service, flag, land, wrong_fragment, urgent, num_failed_logins, root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	**97,42**	**97,40**	**97,43**	**97,39**
20	{duration, service, flag, land, wrong_fragment, urgent, hot, num_failed_logins, root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	97,22	97,06	97,46	96,86
21	{duration, service, flag, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}	97,52	97,33	97,81	97,09
22	{duration, service, flag, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate}	97,01	97,02	96,99	96,98
23	{duration, service, flag, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate}	97,49	97,08	97,99	96,63

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
24	{duration, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate}	97,66	96,92	98,27	96,25
25	{duration, protocol_type, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate}	97,63	97,29	97,91	96,98
26	{duration, protocol_type, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate}	96,80	96,44	97,10	96,11
27	{duration, protocol_type, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate}	97,15	97,14	97,15	97,14

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
28	{duration, protocol_type, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate}	97,21	96,78	97,56	96,39
29	{duration, protocol_type, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	97,22	96,80	97,57	96,42
30	{duration, protocol_type, service, flag, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	97,25	96,87	97,56	96,53

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
31	{duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	96,78	96,23	97,23	95,73
32	{duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	96,83	96,25	97,31	95,72
33	{duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_srv_count, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	95,91	95,48	96,29	95,08

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
34	{duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_srv_count, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	96,12	95,56	96,61	95,04
35	{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, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_srv_count, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	95,48	94,92	95,97	94,40
36	{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, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	96,02	94,87	97,02	93,80

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
37	{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, 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_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	95,66	94,53	96,65	93,47
38	{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, 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_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate}	95,37	93,79	96,74	92,32

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
39	{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, 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}	95,03	93,33	96,52	91,74

Keterangan : Acc. : Accuracy
Sens. : Sensitivity
Spec. : Specificity

Tabel 4.4 Hasil Evaluasi Performansi Subset Fitur pada Kyoto

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
1	{dst_host_srv_serror_rate}	66,67	70,00	60,00	55,00
2	{dst_host_serror_rate, dst_host_srv_serror_rate}	71,43	66,67	77,78	61,12
3	{serror_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	76,74	64,71	84,62	54,76
4	{service, serror_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	85,71	92,50	76,67	68,76
5	{service, source_bytes, serror_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,64	95,00	99,83	92,59
6	{service, source_bytes, serror_rate, srv_serror_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,42	98,18	99,58	97,48
7	{service, source_bytes, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,34	98,35	99,49	97,78

No	Subset	Acc. (%)	Sens. (%)	Spec. (%)	Score
8	{service, source_bytes, destination_bytes, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,55	97,73	99,72	96,74
9	{service,source_bytes, destination_bytes, same_srv_rate, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,17	97,68	99,30	96,87
10	{duration, service, source_bytes, destination_bytes, same_srv_rate, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,75	98,22	99,83	97,42
11	{duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate}	99,70	98,39	99,77	97,70
12	{duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag}	99,79	98,42	99,87	97,70
13	{duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_count, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag}	99,74	98,40	99,84	97,68
14	{duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_count, dst_host_srv_count, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag}	**99,72**	**98,87**	**99,80**	**98,41**

Keterangan : Acc. : Accuracy
Sens. : Sensitivity
Spec. : Specificity

4.2 Hasil Eksperimen Masing-masing Metode pada Setiap *Dataset*

Pada eksperimen ini, metode pembanding B1, B2, dan P akan dievaluasi dengan menggunakan *dataset* pada Tabel 3.2 untuk mengetahui performansi dari masing-masing metode. Hal yang ingin diketahui dari eksperimen ini adalah : (1) apakah metode P dapat memberikan hasil yang terbaik; (2) apakah penggunaan fitur hasil seleksi ini dapat meningkatkan performa dari metode-metode lainnya.

Tabel 4.5 Hasil Eksperimen Metode Pembanding dan Metode yang Diusulkan dengan Menggunakan *Dataset* NSLKDD-6

Metode	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
B1	40,00	0,00	100,00	0,00	5,00
B2	13,33	11,11	16,67	13,33	12,70
P	20,00	11,11	33,33	14,29	4,20

Tabel 4.6 Hasil Eksperimen Metode Pembanding dan Metode yang Diusulkan dengan Menggunakan *Dataset* NSLKDD-8

Metode	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
B1	51,54	69,10	44,57	44,76	5,00
B2	92,60	88,91	94,07	87,23	251,70
P	90,51	81,01	94,28	82,91	339,30

Tabel 4.7 Hasil Eksperimen Metode Pembanding dan Metode yang Diusulkan dengan Menggunakan *Dataset* NSLKDD-19

Metode	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
B1	75,42	59,92	87,88	68,49	5,00
B2	94,28	92,56	95,66	93,52	129,40
P	93,70	90,55	96,23	92,76	29,50

Tabel 4.8 Hasil Eksperimen Metode Pembanding dan Metode yang Diusulkan dengan Menggunakan *Dataset* NSLKDD-P

Metode	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
B1	47,32	46,17	49,02	51,06	5,00
B2	91,33	91,99	90,35	92,66	91,50
P	97,42	97,40	97,43	97,82	426,50

Tabel 4.9 Hasil Eksperimen Metode Pembanding dan Metode yang Diusulkan dengan Menggunakan *Dataset* Kyoto-7

Metode	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
B1	89,37	74,06	90,52	49,22	2,00
B2	95,32	86,03	96,01	71,88	383,00
P	97,47	92,59	97,84	83,61	287,00

Tabel 4.10 Hasil Eksperimen Metode Pembanding dan Metode yang Diusulkan dengan Menggunakan *Dataset* Kyoto-P

Metode	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
B1	95,15	71,59	97,29	71,11	2,00
B2	97,18	94,11	97,46	84,76	116,40
P	99,72	98,87	99,80	98,36	2365,10

Dari Tabel 4.5 terlihat bahwa penggunaan metode B1 pada NSLKDD-6 memberikan nilai *accuracy* tertinggi dibandingkan metode lainnya. Namun jika diperhatikan dengan seksama pada Tabel 4.15, metode B1 memiliki nilai *sensitivity* sama dengan nol yang artinya metode tidak mampu mendeteksi adanya serangan (seluruh data serangan dideteksi sebagai data normal). Hal ini juga dikonfirmasi dengan nilai *F-measure* sama dengan nol. Metode B2 dan P hanya berhasil mendeteksi 1 serangan seperti yang dapat dilihat pada Tabel 4.16 dan Tabel 4.17. Metode B2 dan P pun juga tidak mampu memberikan hasil yang baik, karena satu-satunya serangan yang terdeteksi adalah serangan DoS yang terdeteksi sebagai R2L. Hasil ini berbeda dengan hasil eksperimen pada (Lin dkk., 2015). Perbedaan

hasil ini terjadi karena jumlah data pada dataset yang digunakan terlalu sedikit (dataset NSLKDD-6 terdiri dari 15 data akibat penghilangan data redundan). Jumlah data tersebut tidak cukup untuk membangun model sistem yang sesuai dengan menggunakan 6 fitur {land, urgent, num_failed_logins, num_shells, num_outbound_cmds, is_host_login}.

Hasil eksperimen terhadap NSLKDD-8 seperti pada Tabel 4.6 menunjukkan perbaikan performa dari eksperimen terhadap NSLKDD-6. Secara umum metode B2 mengungguli 2 metode lainnya. Hal ini wajar karena NSLKDD-8 dibentuk berdasarkan hasil seleksi fitur pada (Muchammad dan Ahmad, 2015) sehingga metode B2 dapat memperoleh hasil optimal dengan pengunaan *dataset* ini. Namun yang perlu menjadi catatan, pada eksperimen ini metode B2 tidak berhasil mendeteksi keberadaan serangan U2R (Tabel 4.19). Walaupun secara umum performa metode P berada di bawah metode B2, namun metode P dapat mendeteksi data U2R sebagai serangan (walaupun terjadi kesalahan deteksi sebagai serangan yang lain seperti pada Tabel 4.20). Metode P juga memiliki nilai *specificity* yang sedikit lebih baik (94,28%), yang artinya tingkat penanganan *false alarm* pada metode P di dataset NSLKDD-8 juga lebih baik dari metode B2.

Pada eksperimen dengan NSLKDD-19, performa metode B1 meningkat signifikan jika dibandingkan dengan hasil pada eksperimen dengan NSLKDD-6 dan NSLKDD-8 ditandai dengan nilai accuracy sebesar 75,42%. Secara umum performa metode P berada di bawah metode B2, namun kemampuan metode P dalam mendeteksi data normal, data serangan dos, dan probe dengan benar juga sedikit lebih baik dari metode B2, seperti yang terlihat pada Tabel 4.22 dan Tabel 4.23. Hal ini dicapai dengan jumlah rata-rata pembentukan *cluster* di metode P (29,5 cluster) yang jauh lebih sedikit dari metode B2 (129,40 cluster). Untuk itu, dilakukan eksperimen tambahan untuk mengetahui apakah terdapat nilai ambang radius lebih baik dari yang dihasilkan persamaan 3.2 yang dapat memperbaiki performansi dari metode P pada dataset NSLKDD-19. Hasil dari eksperimen ini dapat dilihat pada bagian 4.3.

Tabel 4.8 memuat hasil eksperimen dengan NSLKDD-P, dapat terlihat metode P mengungguli metode lainnya baik dari nilai *accuracy*, *sensitivity*, *specificity*, dan *F-measure*. Dari hasil eksperimen diketahui kondisi ini diperoleh

pada ambang radius rata-rata 1,39. Dari Tabel 4.24, Tabel 4.25, dan Tabel 4.26 dapat dilihat juga pada *dataset* NSLKDD-P, metode B2 dan P dapat melakukan pendeteksian beberapa serangan U2R, dimana hal ini tidak terjadi pada eksperimen dengan *dataset* NSLKDD-6, NSLKDD-8, dan NSLKDD-19. Dari Tabel 4.26 juga terlihat, bahwa metode P memiliki tingkat kebenaran deteksi data normal atau data serangan (kecuali serangan U2R) yang lebih baik dibandingkan dari hasil pada eksperimen-eksperimen sebelumnya. Untuk serangan U2R, tingkat kebenaran deteksi serangan U2R oleh metode P pada dataset NSLKDD-P hanya sebesar 9,09%, berselisih 1 data dengan hasil metode B2 (sebesar 18,18%). Hal ini kemungkinan terjadi karena jumlah data serangan U2R pada dataset terlalu sedikit (hanya 11 data). Dari hasil eksperimen juga diketahui bahwa metode P melakukan pembentukan *cluster* 4,66 kali lebih banyak dari metode B2 yang berarti metode P melakukan komputasi yang lebih banyak daripada metode B2. Hal lainnya adalah penggunaan metode B1 pada NSLKDD-P ternyata memiliki performansi yang lebih rendah daripada di eksperimen dengan NSLKDD-19.

Hasil eksperimen tiap-tiap metode dengan *dataset* Kyoto-7 dapat dilihat pada Tabel 4.9. Metode B1 hanya membentuk 2 *cluster* karena *dataset* Kyoto yang digunakan merupakan *dataset* dengan 2-*class classification problem* sehingga *clustering* dilakukan dengan k=2 (Lin dkk., 2015). Terlihat bahwa metode P mengungguli metode pembanding B1 dan B2 dari nilai *accuracy*, *sensitivity*, *specificity*, dan *F-measure*. Kemampuan metode P dalam menghindari adanya *false alarm* (kesalahan deteksi data normal sebagai serangan) terlihat lebih baik dari metode B2 ditandai dengan nilai *specificity* metode P (97,84%) yang lebih tinggi dari metode B2 (96,01%) pada dataset Kyoto-7.

Hasil dari eksperimen tiap-tiap metode dengan Kyoto-P seperti pada Tabel 4.10 memperkuat argumentasi bahwa metode P mengungguli metode pembanding B1 dan B2. Dari tabel tersebut terlihat peningkatan dari kemampuan metode P dalam pendeteksian data serangan (*sensitivity*=98,87%) dan pendeteksian data normal (*specificity*=99,80%). Nilai *F-measure* metode P sebesar 98,36% menunjukkan keunggulan metode ini dibandingkan metode pembanding B1 dan B2 dalam mendeteksi data serangan secara menyeluruh (*complete*) dan presisi (*exact*). Dari hasil eksperimen menunjukkan penggunaan metode P memperkecil persentase

data serangan yang terdeteksi normal ataupun data normal yang terdeteksi sebagai serangan. Secara keseluruhan penggunaan metode P dengan *dataset* Kyoto-P memiliki nilai *accuracy* yang lebih baik yaitu sebesar 99,72% jika dibandingkan dengan metode B1 (*accuracy* = 95,15%) dan metode B2 (*accuracy* = 97,18%).

Pengamatan terhadap hasil eksperimen yang dimuat dalam Tabel 4.5, Tabel 4.6, Tabel 4.7, dan Tabel 4.8 juga menunjukkan bahwa hasil terbaik untuk metode B1 dan B2 pada dataset NSLKDD diperoleh pada eksperimen dengan NSLKDD-19, sedangkan hasil terbaik untuk metode P dicapai pada penggunaan subset fitur hasil seleksi fitur seperti pada *dataset* NSLKDD-P. Demikian juga penggunaan hasil seleksi fitur seperti pada dataset Kyoto-P dapat meningkatkan performansi dari metode-metode yang diuji dalam eksperimen ini, seperti yang terlihat pada Tabel 4.9 dan Tabel 4.10.

4.3 Hasil Eksperimen Penentuan Nilai Radius *Cluster* Terbaik

Eksperimen ini dilakukan untuk mengetahui apakah nilai ambang radius yang diperoleh melalui persamaan 3.1 dapat memberikan hasil yang terbaik. Untuk itu dilakukan eksperimen dengan menggunakan metode P pada beberapa dataset dengan menggunakan beberapa nilai *radius cluster*.

4.3.1. Hasil Eksperimen Nilai Ambang Radius Metode P pada Dataset NSLKDD-P

Tabel 4.11 Hasil Eksperimen Metode P dengan *Dataset* NSLKDD-P dengan Beberapa Nilai Ambang Radius

Ambang Radius	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
1,00	98,36	98,17	98,63	98,61	4198,20
1,10	97,67	97,35	98,13	98,03	564,70
1,20	97,55	97,25	98,00	97,93	518,20
1,30	97,47	97,21	97,85	97,86	484,40
1,39	97,42	97,40	97,43	97,82	426,50
1,40	97,14	97,19	97,06	97,59	448,30
1,50	96,33	96,07	96,69	96,89	94,80

Dari Tabel 4.11 terlihat pada nilai ambang radius 1.0, tercapai nilai *accuracy*, *specificity*, dan *specificity* yang lebih baik. Namun jumlah rata-rata

cluster yang terbentuk pada proses clustering lebih besar jika dibandingkan saat penggunaan nilai ambang radius rata-rata sesuai persamaan 3.1 (ambang radius = 1,39). Pengunaan nilai ambang radius 1,10, 1,20, dan 1,30 memberikan hasil yang sedikit lebih baik daripada nilai ambang 1,39 dengan konsekuensi jumlah *cluster* yang terbentuk menjadi lebih banyak. Demikian juga dengan nilai ambang radius 1,50 walaupun jumlah rata-rata cluster yang terbentuk lebih sedikit dari nilai ambang 1,39, namun jika dibandingkan nilainya dengan menggunakan persamaan 3.5, 3.6, dan 3,7 hasil tersebut masih belum mengungguli dari hasil yang diperoleh dari penggunaan nilai ambang radius rata-rata 1,39. Berdasarkan hasil ini, penulis berpendapat bahwa penggunaan nilai ambang radius terbaik untuk metode P pada dataset NSLKDD-P adalah 1,39 sesuai persamaan 3.2.

4.3.2. Hasil Eksperimen Nilai Ambang Radius Metode P pada Dataset Kyoto-P

Tabel 4.12 Hasil Eksperimen Metode P dengan *Dataset* Kyoto-P dengan Beberapa Nilai Ambang Radius

Ambang Radius	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
0,70	99,74	99,01	99,80	98,44	2593,80
0,80	99,73	98,99	99,79	98,38	2558,80
0,90	99,72	98,96	99,79	98,34	2505,30
0,99	99,72	98,87	99,80	98,36	2365,10
1,00	99,72	98,89	99,79	98,32	2211,00
1,10	99,63	97,99	99,78	97,76	319,30

Dari beberapa nilai ambang radius seperti pada Tabel 4.12, nilai *accuracy*, *sensitivity*, dan *specificity* tertinggi dicapai saat nilai ambang radius adalah 0,70, sedangkan untuk nilai ambang radius 0,9, 0,99, dan 1.00 memiliki nilai accuracy yang sama yaitu 99,72. Dari ketiga nilai tersebut, penggunaan ambang radius 1,0 menghasilkan jumlah rata-rata cluster yang lebih sedikit. Terlihat bahwa nilai ambang radius 0,99 yang dihitung secara otomatis dengan persamaan 3.2 memberikan hasil yang tidak jauh berbeda dengan ambang radius terbaik 1,0. Dari

eksperimen ini dapat disimpulkan bahwa penghitungan ambang radius sesuai persamaan 3.2 dapat digunakan untuk meningkatkan performansi dari metode yang diajukan.

4.3.3. Hasil Eksperimen Nilai Ambang Radius Metode P pada Dataset NSLKDD-19

Pada bagian 4.2, eksperimen metode P pada dataset NSLKDD-19 dengan menggunakan nilai ambang radius berupa rata-rata jarak antar data (sesuai persamaan 3.2) ternyata belum memberikan hasil terbaik. Untuk itu dilakukan eksperimen tambahan dengan menggunakan beberapa nilai ambang radius untuk mengetahui nilai ambang radius yang dapat meningkatkan hasil dari metode P. Hasil dari eksperimen ini dapat dilihat pada Tabel 4.13. Dari hasil tersebut dapat diketahui dengan pengubahan nilai ambang radius dari nilai rata-rata radius (2,08) ke nilai 2,0, dapat meningkatkan nilai *accuracy*, *sensitivity*, *specificity*, dan *F-measure* metode P jika dibandingkan dengan metode B2 dengan jumlah rata-rata pembentukan cluster yang lebih sedikit (75,4 cluster).

Tabel 4.13 Hasil Eksperimen Metode P dengan *Dataset* NSLKDD-19 dengan Beberapa Nilai Ambang Radius

Ambang Radius	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
1,80	95,45	93,67	96,87	94,83	170,50
1,90	95,34	93,68	96,67	94,71	147,10
2,00	94,96	92,94	96,58	94,26	75,40
2,08	93,70	90,55	96,23	92,76	29,50

4.4 Hasil Eksperimen Pembandingan Penghitungan Jarak dari Metode pembanding B2 dan Metode yang diusulkan P pada NSLKDD-P

Hasil eksperimen pembandingan penghitungan jarak antara metode B2 dan metode P terhadap dataset NSLKDD-P disajikan seperti pada Tabel 4.14. Eksperimen #1, #2, #3 adalah pengujian implementasi dari metode (Muchammad dan Ahmad, 2015), berturut-turut : tanpa menggunakan penghitungan jarak ke *sub-centroid*, dengan menggunakan penghitungan jarak ke *sub-centroid*, dan menggunakan penghitungan jarak ke *sub-medoid* untuk menggantikan

penghitungan jarak ke *sub-centroid*. Eksperimen #4, #5, #6 adalah pengujian dari metode yang diajukan, berturut-turut : tanpa menggunakan penghitungan jarak ke *sub-medoid*, dengan menggunakan penghitungan jarak ke *sub-centroid* untuk menggantikan penghitungan jarak ke *sub-medoid*, dan menggunakan penghitungan jarak ke *sub-medoid*.

Dari hasil eksperimen #1 dan #4 pada dataset NSLKDD-P terlihat bahwa penggunaan *threshold* berupa *radius* (seperti pada eksperimen #4) memberikan nilai *accuracy*, *sensitivity*, *specificity*, dan *F-measure* yang lebih baik. Dominasi performansi tersebut juga terlihat pada eksperimen #5 dan #6. Dari Tabel 4.14 juga terlihat bahwa penggunaan jarak ke *sub-medoid* memberikan nilai performansi yang sedikit lebih baik daripada penggunaan jarak ke *sub-centroid*, sebagai contoh terlihat pada hasil eksperimen #6 yang memiliki *accuracy* 0,6% lebih baik daripada hasil eksperimen #5. Walaupun demikian, seperti halnya pada hasil eksperimen pada Tabel 4.8, metode yang diusulkan P melakukan pembentukan *cluster* yang lebih banyak daripada metode B2 dalam pemrosesan data *training*, hal ini merupakan indikasi bahwa metode P akan melakukan proses komputasi yang lebih banyak daripada metode B2 untuk melakukan peningkatan performansi ini.

Tabel 4.14 Hasil Eksperimen Pembandingan Penghitungan Jarak dari Metode pembanding B2 dan Metode yang diusulkan P pada NSLKDD-P

Eksperimen ke -	*Accuracy* (%)	*Sensitivity* (%)	*Specificity* (%)	*F-measure* (%)	Jumlah Rata-rata *Cluster*
#1	91,44	92,16	90,38	92,76	91,50
#2	91,33	91,99	90,35	92,66	91,50
#3	91,46	92,16	90,44	92,78	91,50
#4	97,43	97,36	97,53	97,83	426,50
#5	97,38	97,45	97,28	97,79	426,50
#6	97,42	97,40	97,43	97,82	426,50

Keterangan : #1) metode B2 tanpa melakukan penghitungan jarak ke *sub-centroid*
#2) metode B2 dengan penghitungan jarak ke *sub-centroid*
#3) metode B2 dengan penghitungan jarak ke *sub-medoid*
#4) metode P tanpa melakukan penghitungan jarak ke *sub-medoid*
#5) metode P dengan penghitungan jarak ke *sub-centroid*
#6) metode P dengan penghitungan jarak ke *sub-medoid*

4.5 Hasil Deteksi pada Eksperimen

Bagian berikut memuat hasil pendeteksian label data yang diperoleh dari eksperimen pada bagian 4.2 tentang eksperimen metode B1, metode B2, dan metode P pada *dataset* NSLKDD-6, NSLKDD-8, NSLKDD-19, NSLKDD-P, Kyoto-7, dan Kyoto-P. Hasil tersebut disajikan dalam tabel yang memuat jumlah data aktual yang berhasil dideteksi oleh masing-masing metode untuk setiap label pada *dataset* tersebut.

Tabel 4.15 Hasil Deteksi Metode Pembanding B1 dengan Menggunakan *Dataset* NSLKDD-6

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	6	0	0	0	0
DoS	2	0	0	0	0
R2L	3	0	0	0	0
Probe	2	0	0	0	0
U2R	2	0	0	0	0

Tabel 4.16 Hasil Deteksi oleh Metode Pembanding B2 dengan Menggunakan *Dataset* NSLKDD-6

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	1	1	3	1	0
DoS	1	0	1	0	0
R2L	3	0	0	0	0
Probe	2	0	0	0	0
U2R	2	0	0	0	0

Tabel 4.17 Hasil Deteksi oleh Metode Yang Diajukan P dengan Menggunakan *Dataset* NSLKDD-6

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	2	1	3	0	0
DoS	1	0	1	0	0
R2L	3	0	0	0	0
Probe	2	0	0	0	0
U2R	2	0	0	0	0

Tabel 4.18 Hasil Deteksi oleh Metode Pembanding B1 dengan Menggunakan *Dataset* NSLKDD-8

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	5610	2226	32	4720	0
DoS	1047	1216	6	1641	0
R2L	68	16	1	25	0
Probe	421	215	13	317	0
U2R	8	2	0	1	0

Tabel 4.19 Hasil Deteksi oleh Metode Pembanding B2 dengan Menggunakan *Dataset* NSLKDD-8

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	11841	404	66	249	28
DoS	328	3551	5	24	2
R2L	46	4	54	3	3
Probe	169	12	8	776	1
U2R	11	0	0	0	0

Tabel 4.20 Hasil Deteksi oleh Metode Yang Diajukan P dengan Menggunakan *Dataset* NSLKDD-8

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	11868	505	10	199	6
DoS	634	3253	2	21	0
R2L	54	3	45	6	2
Probe	252	18	10	685	1
U2R	9	0	1	1	0

Tabel 4.21 Hasil Deteksi oleh Metode Pembanding B1 dengan Menggunakan *Dataset* NSLKDD-19

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	11606	1146	108	345	1
DoS	3630	4583	8	469	0
R2L	157	37	9	6	0
Probe	462	587	8	653	0
U2R	7	4	0	0	0

Tabel 4.22 Hasil Deteksi oleh Metode Pembanding B2 dengan Menggunakan *Dataset* NSLKDD-19

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	12633	108	156	306	3
DoS	496	8073	7	114	0
R2L	115	2	91	1	0
Probe	171	334	12	1193	0
U2R	8	0	2	1	0

Tabel 4.23 Hasil Deteksi oleh Metode Yang Diajukan P dengan Menggunakan *Dataset* NSLKDD-19

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	12708	217	47	232	2
DoS	483	8085	1	121	0
R2L	131	1	71	6	0
Probe	380	134	2	1194	0
U2R	10	0	1	0	0

Tabel 4.24 Hasil Deteksi oleh Metode Pembanding B1 dengan Menggunakan *Dataset* NSLKDD-P

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	3439	2645	8	920	4
DoS	4926	3338	10	665	0
R2L	38	13	0	19	0
Probe	587	318	3	389	1
U2R	3	2	0	6	0

Tabel 4.25 Hasil Deteksi oleh Metode Pembanding B2 dengan Menggunakan *Dataset* NSLKDD-P

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	6339	88	217	353	19
DoS	657	7950	11	321	0
R2L	42	0	21	2	5
Probe	121	50	1	1126	0
U2R	6	0	2	1	2

Tabel 4.26 Hasil Deteksi oleh Metode Yang Diajukan P dengan Menggunakan *Dataset* NSLKDD-P

	Hasil Deteksi				
Aktual	Normal	DoS	R2L	Probe	U2R
Normal	6836	105	23	50	2
DoS	140	8733	1	65	0
R2L	32	4	34	0	0
Probe	90	21	3	1184	0
U2R	6	0	3	1	1

Tabel 4.27 Hasil Deteksi oleh Metode Pembanding B1 dengan Menggunakan *Dataset* Kyoto-7

	Hasil Deteksi	
Aktual	Normal	Serangan (attack)
Normal	53911	5649
Serangan (attack)	1155	3298

Tabel 4.28 Hasil Deteksi oleh Metode Pembanding B2 dengan Menggunakan *Dataset* Kyoto-7

	Hasil Deteksi	
Aktual	Normal	Serangan (attack)
Normal	57184	2376
Serangan (attack)	622	3831

Tabel 4.29 Hasil Deteksi oleh Metode Yang Diajukan P dengan Menggunakan *Dataset* Kyoto-7

	Hasil Deteksi	
Aktual	Normal	Serangan (attack)
Normal	58273	1287
Serangan (attack)	330	4123

Tabel 4.30 Hasil Deteksi oleh Metode Pembanding B1 dengan Menggunakan *Dataset* Kyoto-P

	Hasil Deteksi	
Aktual	Normal	Serangan (attack)
Normal	61549	1714
Serangan (attack)	1636	4122

Tabel 4.31 Hasil Deteksi oleh Metode Pembanding B2 dengan Menggunakan *Dataset* Kyoto-P

	Hasil Deteksi	
Aktual	Normal	Serangan (attack)
Normal	61654	1609
Serangan (attack)	339	5419

Tabel 4.32 Hasil Deteksi oleh Metode Yang Diajukan P dengan Menggunakan *Dataset* Kyoto-P

	Hasil Deteksi	
Aktual	Normal	Serangan (attack)
Normal	63138	125
Serangan (attack)	65	5693

BAB 5
KESIMPULAN

Berdasarkan hasil eksperimen yang telah dilakukan, diperoleh kesimpulan sebagai berikut :

a. Pemilihan subset fitur yang tepat dapat meningkatkan performa dari sistem deteksi intrusi. Subset fitur tersebut dapat diperoleh dengan melakukan seleksi fitur secara bertahap kepada dataset. Subset fitur terbaik adalah subset fitur yang memiliki nilai rata-rata *sensitivity* dan *specificity* terbesar dan memiliki nilai yang berimbang.
b. Fitur-fitur signifikan dari dataset NSLKDD yang diperoleh dari hasil eksperimen ini adalah 19 fitur yang terdiri dari {duration, service, flag, land, wrong_fragment, urgent, num_failed_logins, root_shell, su_attempted, num_file_creations, num_shells, num_access_files, is_guest_login, count, srv_count, diff_srv_rate, srv_diff_host_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate}. Sedangkan pada dataset Kyoto, seluruh fiturnya adalah fitur signifikan yang terdiri dari {duration, service, source_bytes, destination_bytes, count, same_srv_rate, serror_rate, srv_serror_rate, dst_host_count, dst_host_srv_count, dst_host_same_src_port_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag, label}. Penggunaan 19 fitur dari dataset NSLKDD memberikan performa deteksi terbaik untuk metode yang diajukan pada penelitian ini. Dari penelitian ini juga disimpulkan bahwa penggunaan ke-14 fitur dari dataset Kyoto dapat memberikan performa deteksi yang lebih baik untuk semua metode yang diuji.
c. Pembatasan ukuran *cluster* dengan menggunakan ambang batas berupa radius *cluster* dapat meningkatkan performa dari proses deteksi serangan. Nilai tersebut dapat dihitung sebagai jarak rata-rata antara data pada *dataset* sehingga tidak memerlukan masukan dari pengguna. Hal ini terlihat pada hasil eksperimen pada bagian 4.4.
d. Penggunaan jarak ke sub-medoid dapat meningkatkan performa deteksi sebagaimana terlihat pada hasil eksperimen pada bagian 4.4.

Sebagai tindak lanjut penelitian ini, beberapa hal yang dapat disarankan untuk penelitian berikutnya adalah :

a. Penggunaan teknik yang lebih efektif untuk melakukan penyeleksian fitur berdasarkan subset fitur. Proses seleksi fitur yang lebih cepat meningkatkan kecepatan dari model deteksi intrusi untuk beradaptasi dengan bermacam-macam dataset.
b. Penggunaan algoritma clustering yang lebih baik untuk melakukan partisi berdasarkan medoid *cluster*. Pada penelitian ini, proses pembentukan sub-centroid masih menggunakan algoritma PAM yang kurang efektif untuk digunakan pada dataset berukuran besar.

DAFTAR PUSTAKA

Agrawal, Shikha dan Agrawal, Jitendra (2015), "Survey on Anomaly Detection using Data Mining Techniques", *Procedia Computer Science*, Vol. 60, hal. 708-713.

Chae, Hee-su, Jo, Byung-oh, Choi, Sang-Hyun, dan Park, Twae-kyung (2015), "Feature Selection for Intrusion Detection using NSL-KDD", *Recent Advances in Computer Science*, hal. 978-960.

Chapelle, O., Schölkopf, B. dan Zien, A., eds. (2006), *Semi-Supervised Learning*. Cambridge: MIT Press.

Duque, Solane dan Omar, Mohd. Nizam bin (2015), "Using Data Mining Algorithms for Developing a Model for Intrusion Detection System (IDS)", *Procedia Computer Science*, Vol. 61, hal. 46-51.

Guyon, Isabelle dan Elisseeff, André (2003), "An Introduction to Variable and Feature Selection", *The Journal of Machine Learning Research*, Vol. 3, hal. 1157–1182.

Hall, Mark, Frank, Eibe, Holmes, Geoffrey, Pfahringer, Benhard, Reutemann, Peter, dan Witten, Ian H. (2009), "The WEKA Data Mining Software: An Update", *ACM SIGKDD Explorations Newsletter*, Vol. 11, No. 1, hal. 10-18.

Hall, Mark A. dan Holmes, Geoffrey (2003), "Benchmarking Attribute Selection Techniques for Discrete Class Data Mining", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 15, No. 6, hal. 1437-1447.

Han, Li (2011), "Using a Dynamic K-means Algorithm to Detect Anomaly Activities", *2011 Seventh International Conference on Computational Intelligence and Security (CIS)*, IEEE, Hainan, hal. 1049-1052.

Han, Eui-Hong (Sam) dan Karpys, George (2000), "Centroid-based document classification: Analysis and experimental results", *European Conference on Principles of Data Mining and Knowledge Discovery*, Springer Berlin Heidelberg, Lyon, hal. 424-431.

Hettich, S. dan Bay, S. D., 1999. *KDD Cup 1999 Data*. [Online] University of California, Department of Information and Computer Science Tersedia di: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

Kaufman, Leonard dan Rousseeuw, Peter J. (1990), *Finding Groups in Data: An Introduction to Cluster Analysis*. New Jersey: John Wiley & Sons, Inc.

Lin, Wei-Chao, Ke, Shih-Wen, dan Tsai, Chih-Fong (2015), "CANN: An Intrusion Detection System Based on Combining Cluster Centers and Nearest Neighbors", *Knowledge-Based Systems*, Vol. 78, hal. 13-21.

Muchammad, Kharisma dan Ahmad, Tohari (2015), "Detecting Intrusion Using Recursive Clustering and Sum of Log Distance to Sub-centroid", *Procedia Computer Science*, Vol. 72, hal. 446-452.

Singh, Raman, Kumar, Harish, dan Singla, R.K. (2015), "An Intrusion Detection System Using Network Traffic Profiling and Online", *Expert Systems with Applications*, Vol. 46, No. 22, hal. 8609-8624.

Sommer, Robin dan Paxson, Vern (2010), "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection", *2010 IEEE Symposium on Security and Privacy (SP)*, IEEE, Oakland, hal. 305-316.

Song, Jungsuk, Takakura, Hiroki, Okabe, Yasuo, Eto, Masashi, Inoue, Daisuke, dan Nakao, Koji (2011), "Statistical Analysis of Honeypot Data and Building of Kyoto 2006+ Dataset for NIDS Evaluation", *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, ACM, Salzburg, hal. 29-36.

Tavallaee, M., Bagheri, E., Lu, Wei, dan Ghorbani, A. A. (2009), "A Detailed Analysis of the KDD Cup 99 Data Set", *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA 2009)*, IEEE, Ottawa, hal. 2-6.

Tsai, Chih-Fong dan Lin, Chia-Ying (2010), "A Triangle Area Based Nearest Neighbors Approach to Intrusion Detection", *Pattern Recognition*, Vol. 43, No. 1, hal. 222-229.

BIOGRAFI PENULIS

Penulis, Indera Zainul Muttaqien, lahir di Surabaya pada tanggal 25 November 1981. Putra kedua dari pasangan Achmad Wasil (alm.) dan Maimunah Maschab. Penulis menempuh pendidikan SD (1987-1993), SMP (1993-1996), dan SMA (1996-1999) di sekolah Taruna Dra. Zulaeha. Pendidikan S1 ditempuh di Jurusan Teknik Informatika Institut Teknologi Bandung (1999-2004).

Dalam menempuh pendidikan S1, penulis mengambil bidang minat Jaringan dan pada pendidikan S2 penulis mengambil bidang minat Komputasi Berbasis Jaringan. Saat ini penulis bekerja di seksi Sistem Informasi, bidang Kelembagaan dan Sistem Informasi Kopertis Wilayah VII. Penulis dapat dihubungi melalui email inderazainul@gmail.com.

[Halaman ini sengaja dikosongkan]