Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – KI141502

KLASIFIKASI DATA EEG UNTUK MENDETEKSI KEADAAN TIDUR DAN BANGUN MENGGUNAKAN AUTOREGRESSIVE (AR) MODEL DAN SUPPORT VECTOR MACHINE (SVM)

YUNAN HELMI MAHENDRA
NRP 5112 100 077

Dosen Pembimbing I
Prof.Ir. Handayani Tjandrasa, M.Sc., Ph.D.

Dosen Pembimbing II
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

TUGAS AKHIR – KI141502

KLASIFIKASI DATA EEG UNTUK MENDETEKSI KEADAAN TIDUR DAN BANGUN MENGGUNAKAN AUTOREGRESSIVE (AR) MODEL DAN SUPPORT VECTOR MACHINE (SVM)

YUNAN HELMI MAHENDRA
NRP 5112 100 077

Dosen Pembimbing I
Prof.Ir. Handayani Tjandrasa, M.Sc., Ph.D.

Dosen Pembimbing II
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

(Halaman ini sengaja dikosongkan)

FINAL PROJECT – KI141502

CLASSIFICATION OF EEG DATA TO DETECT SLEEP AND AWAKE CONDITION USING AUTOREGRESSIVE (AR) MODEL AND SUPPORT VECTOR MACHINE (SVM)

Yunan Helmi Mahendra
NRP 5112 100 077

First Supervisor
Prof.Ir. Handayani Tjandrasa, M.Sc., Ph.D.

Second Supervisor
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

INFORMATICS DEPARTMENT
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

KLASIFIKASI DATA EEG UNTUK MENDETEKSI KEADAAN TIDUR DAN BANGUN MENGGUNAKAN AUTOREGRESSIVE (AR) MODEL DAN SUPPORT VECTOR MACHINE (SVM)

TUGAS AKHIR

Diajukan untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
YUNAN HELMI MAHENDRA
NRP. 5112 100 077

Disetujui oleh Pembimbing Tugas Akhir:

1. Prof.Ir. Handayani Tjandrasa, M.Sc.Eng.Ph.D. ..... (Pembimbing I)
   NIP. 194908231976032001
2. Dr. Eng. Chastine Fatichah, S.Kom, M.Kom. ..... (Pembimbing II)
   NIP. 197512202001122002

SURABAYA
JUNI, 2016

(Halaman ini sengaja dikosongkan)

KLASIFIKASI DATA EEG UNTUK MENDETEKSI KEADAAN TIDUR DAN BANGUN MENGGUNAKAN AUTOREGRESSIVE (AR) MODEL DAN SUPPORT VECTOR MACHINE (SVM)

Nama Mahasiswa	**: Yunan Helmi Mahendra**
NRP	**: 5112 100 077**
Jurusan	**: Teknik Informatika FTIf – ITS**
Dosen Pembimbing I	**: Prof.Ir. Handayani Tjandrasa, M.Sc., Ph.D.**
Dosen Pembimbing II	**: Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.**

Abstrak

Tidur merupakan kebutuhan dasar manusia. Fungsi tidur antara lain untuk pemulihan energi, pemugaran otak, meningkatkan fungsi imunitas. Sehingga orang yang memiliki gangguan tidur akan menyebabkan menurunnya sistem kekebalan tubuh. Salah satu gangguan tidur yang cukup berbahaya adalah narkolepsi, yaitu gangguan tidur kronis yang ditandai dengan rasa kantuk yang luar biasa di siang hari dan serangan tidur yang terjadi secara tiba-tiba. Salah satu metode dokter untuk mendiagnosis penyakit narkolepsi adalah dengan melihat aktivitas gelombang otak (melalui sinyal EEG) pasien pada saat tertidur. Tugas akhir ini bertujuan untuk mengembangkan perangkat lunak yang dapat mengklasifikasikan keadaan tidur dan bangun melalui sinyal EEG menggunakan metode Autoregressive (AR) Model sebagai metode ekstraksi fitur dan Support Vector Machine (SVM) sebagai metode klasifikasi..

Dataset EEG yang digunakan tersedia di Physionet. Pertama-tama data EEG yang menjadi masukan dilakukan normalisasi dan filtering. Proses filtering dilakukan untuk membagi data menjadi 3 subband yaitu theta, alpha, dan beta. Setelah itu pada masing-masing subband dilakukan tahap

ekstraksi fitur menggunakan Autoregressive Model. Hasil estimasi koefisien AR model digunakan sebagai fitur. Metode yang digunakan untuk mengestimasi koefisien AR model yaitu metode Yule-Walker dan metode Burg. Dataset dibagi menjadi data latih dan data uji menggunakan 10-fold cross validation. Data training digunakan untuk membuat SVM Model. SVM Model digunakan untuk mengklasifikasikan data testing sehingga menghasilkan keluaran label 1 untuk tidur dan label 0 untuk bangun. Untuk menentukan kelas final dilakukan majority vote dari hasil klasifikasi masing-masing subband.

Performa sistem diperoleh dengan menghitung akurasi, presisi, dan sensitivitas pada setiap skenario uji coba. Skenario uji coba yang dilakukan antara lain dengan memvariasikan order AR, fungsi kernel, dan parameter C pada SVM. Dari hasil uji coba yang dilakukan, metode Yule-Walker menghasilkan rata-rata akurasi 80.60%, presisi 78.19%, dan sensitivitas 77.56%. Metode Burg menghasilkan akurasi 94.01%, presisi 95.70%, dan sensitivitas 93.39%. Hasil tersebut menunjukkan metode Burg memiliki performa lebih baik dibandingan dengan metode Yule-Walker.

Kata Kunci: Autoregressive Model, Elektroensefalografi, Support Vector Machine, Tidur.

CLASSIFICATION OF EEG DATA TO DETECT SLEEP AND AWAKE CONDITION USING AUTOREGRESSIVE (AR) MODEL AND SUPPORT VECTOR MACHINE (SVM)

Student Name	**: Yunan Helmi Mahendra**
NRP	**: 5112 100 077**
Major	**: Teknik Informatika FTIf – ITS**
Supervisor I	**: Prof.Ir. Handayani Tjandrasa, M.Sc., Ph.D.**
Supervisor II	**: Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.**

Abstract

Sleep is basic human necessity. Sleep function is for energy recovery, restoration of the brain, and boost immunity. People who have sleep disorder will lead to decreased immune system. One of the dangerous sleep disorders is narcolepsy. Narcolepsy is a chronic sleep disorder characterized by excessive sleepiness during the day and sleep attacks that occur suddenly. Usually, doctors diagnose narcolepsy by looking at brain wave activity (through EEG signal) of patients during sleep. The purpose of this final project is to develop software for classification of EEG data to detect sleep and awake condition through EEG signals using autoregressive (AR) model as a feature extraction method and support vector machine (SVM) as a classification method.

EEG sleep dataset was provided by Physionet. First, EEG data are normalized and filtered. The aim of filtering process is to split data into three subbands: theta, alpha, beta. For each subband, feature extraction is done using by autoregressive (AR) model. The results of estimated coefficients of AR model are used as features. The methods used to estimate coefficients of AR model are Yule-Walker method and Burg method. Dataset is divided into training data and testing data using 10-fold cross validation.

Training data is used to create SVM model. SVM Model is used to classify testing data which will produce output label 1 for sleep and label 0 for awake. Final class will be determined by majority voting of the classification results of each subband.

System performance is obtained by calculating accuracy, precision, and sensitivity in each experiments. Experiments were performed by varying order AR model, kernel function of SVM, and parameter C of SVM. The average of accuracy, precision, and sensitivity that obtained when using Yule-Walker method respectively are 80.60%, 78.19%, and 77.56%. The average of accuracy, precision, and sensitivity that obtained when using Burg method respectively are 94.01%, 95.70%, and 93.39%. These result showed that Burg method outperfomed Yule-Walker method.

Keywords: Autoregressive Model, Electroencephalography, Sleep, Support Vector Machine.

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Alhamdulillahi rabbil ‘alamin, segala puji hanya bagi Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul: *“Klasifikasi Data EEG untuk Mendeteksi Keadaan Tidur dan Bangun Menggunakan Autoregressive (AR) Model dan Support Vector Machine (SVM)”*yang merupakan salah satu syarat dalam menempuh ujian sidang guna memperoleh gelar Sarjana Komputer dengan tepat waktu.

Dalam pelaksanaan dan pembuatan tugas akhir ini penulis mendapatkan banyak bantuan dari berbagai pihak. Pada kesempatan ini, ucapan terima kasih pennulis berikan kepada:

1. Allah SWT, karena atas limpahan rahmat-Nya, penulis diberikan kemudahan dan kelancaran dalam mengerjakan tugas akhir ini.
2. Bapak Makhin, Ibu Endang Sri Purworini, dan Arzak Primadhana selaku orangtua dan kakak kandung penulis yang selalu memberikan dukungan doa, moral, dan material sehingga penulis dapat menyelesaikan tugas akhir ini.
3. Ibu Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D. dan Ibu Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. selaku dosen pembimbing yang telah memberikan motivasi, nasehat dan bimbingan dalam menyelesaikan tugas akhir ini.
4. Bapak Ir. FX. Arunanto, M.Sc. selaku dosen wali penulis yang telah memberikan arahan, masukan dan motivasi selama perkuliahan.
5. Bapak Dr. Eng. Darlis Herumurti, S.Kom., M.Kom. selaku kepala jurusan Teknik Informatika ITS dan segenap dosen dan karyawan Teknik Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa studi di ITS.

6. Sahabat penulis yang tidak dapat disebutkan satu per satu yang selalu membantu, menghibur, menjadi tempat bertukar ilmu dan berjuang bersama-sama penulis.

Surabaya, Juni 2016

Yunan Helmi Mahendra

DAFTAR ISI

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

(Halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

Pada bab ini akan dibahas mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan tugas akhir. Diharapkan dari penjelasan dalam bab ini gambaran tugas akhir secara umum dapat dipahami.

1.1 Latar Belakang

Tidur merupakan kebutuhan dasar manusia. Fungsi tidur antara lain untuk pemulihan energi, pemugaran otak, meningkatkan fungsi imunitas. Beberapa ahli berpendapat bahwa fungsi tidur sebagai proses detoksifikasi (penetralan) toksin atau racun yang terakumulasi dalam tubuh. Akumulasi toksin inilah yang menyebabkan timbulnya rasa kantuk sehingga memicu seseorang untuk tidur. Penggunaan energi menurun sekitar 15-20% dan konsumsi oksigen menurun saat seseorang tertidur. Orang yang mengalami gangguan tidur, akan mengakibatkan konsentrasi menjadi menurun, daya tahan melemah, juga memicu terserangnya berbagai penyakit berbahaya seperti *kanker*, *stroke* [1].

Selama tertidur, seseorang mengistirahatkan beberapa organ tubuhnya, salah satunya adalah otak. Sehingga terdapat perbedaan aktivitas gelombang otak pada orang yang terjaga (bangun) dengan orang yang tidur. Perbedaan tersebut dapat diidentifikasi menggunakan *electroencephalography*. *Electroencephalography* atau biasa disebut dengan EEG adalah teknik perekaman aktivitas listrik pada kulit kepala yang dihasilkan oleh struktur otak yang diambil dengan logam elektroda dan media konduktif. EEG mengacu pada rekaman aktivitas listrik spontan pada otak selama periode tertentu [2].

Selama ini metode yang digunakan untuk mendeteksi keadaan tidur melalui sinyal EEG masih dilakukan secara manual oleh seorang ahli [3]. Hal tersebut membutuhkan banyak biaya dan waktu. Oleh karena itu dibutuhkan suatu perangkat lunak untuk mendeteksi secara otomatis keadaan tidur dan bangun (terjaga) melalui sinyal EEG.

1.2 Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana mengembangkan perangkat lunak untuk mengklasifikasikan keadaan tidur dan bangun pada dataset yang tersedia dengan menggunakan metode *Autoregressive Model* dan *Support Vector Machine*?
2. Bagaimana performa dari perangkat lunak yang telah dibuat?

1.3 Batasan Masalah

Permasalahan yang dibahas di dalam tugas akhir ini memiliki beberapa batasan masalah sebagai berikut:

1. Aplikasi ini dikembangkan menggunakan matlab.
2. Dataset yang digunakan untuk data latih dan data uji adalah data EEG *Sleep-EDF Database, v1 [deprecated, use sleep-edfx] (sleep-edf)* dari Physionet (http://www.physionet.org/cgi-bin/atm/ATM), *channel Fpz-Oz*.
3. Pengklasifikasian data sinyal EEG berdasarkan keadaan tidur (*sleep*) atau bangun (*awake*).

1.4 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut:

1. Merancang dan membangun perangkat lunak yang dapat mengklasifikasikan keadaan tidur dan bangun pada dataset yang tersedia dengan metode *Autoregressive Model dan Support Vector Machine*.
2. Mengetahui performa dari perangkat lunak yang telah dibuat.

1.5 Manfaat

Dengan dibuatnya tugas akhir ini diharapkan dapat memberikan manfaat pada dunia kedokteran untuk membedakan

apakah seseorang sedang dalam keadaan tidur atau bangun, yang akan membantu dokter dalam mendiagnosa dan mengobati gangguan tidur *narcolepsy* (gangguan neurologis kronis, mempengaruhi bagian otak yang mengatur kapan seseorang tidur dan terjaga) [4].

1.6 Metodologi

Pembuatan tugas akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1.6.1. Penyusunan Proposal Tugas Akhir

Tahap awal yang dilakukan dalam pengerjaan tugas akhir ini adalah menyusun proposal tugas akhir. Pada proposal, diajukan gagasan pembuatan perangkat lunak untuk melakukan klasifikasi keadaan tidur dan bangun dengan dataset sinyal EEG.

1.6.2. Studi Literatur

Pada tahap ini dilakukan pencarian, pengumpulan, penyaringan, pemahaman, dan pembelajaran literatur yang berhubungan dengan metode yang digunakan yaitu *Autoregressive Model* dan *Support Vector Machine*. Literatur yang digunakan meliputi: buku referensi, jurnal, dan dokumentasi internet.

1.6.3. Implementasi dan pembuatan perangkat lunak

Pada tahap ini dilakukan implementasi perangkat lunak sesuai dengan rancangan perangkat lunak yang dibuat. Implementasi dilakukan pada suatu perangkat lunak yaitu Matlab R2015a dan *toolbox* EDF Browser.

1.6.4. Pengujian dan Evaluasi

Pada tahap ini dilakukan uji coba terhadap perangkat lunak yang telah dibuat untuk mengetahui kemampuan algoritma yang digunakan. Uji coba menggunakan dataset *Sleep-EDF Database, v1 [deprecated, use sleep-edfx] (sleep-edf)* dari (http://www.physionet.org/cgi-bin/atm/ATM) yang berisi dataset

sinyal EEG dari delapan subjek yang berbeda. Uji coba dilakukan dengan membandingkan metode Yule Walker dan Burg untuk mengestimasi koefisien AR Model pada proses ekstraksi fitur. Tahap evaluasi dilakukan dengan menghitung akurasi, presisi, sensitivitas. Evaluasi menggunakan metode *ten fold cross validation*.

1.6.5. Penyusunan Buku Tugas Akhir

Tahap ini merupakan tahap dokumentasi tugas akhir. Buku tugas akhir yang disusun berisi dasar teori, perancangan, implementasi, serta hasil uji coba dan evaluasi dari perangkat lunak yang dibangun.

1.7 Sistematika Penulisan

Buku tugas akhir ini disusun dengan sistematika penulisan sebagai berikut:

1. Bab I. Pendahuluan
   Bab pendahuluan berisi latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan buku tugas akhir.
2. Bab II. Tinjauan Pustaka
   Bab tinjauan pustaka berisi penjelasan mengenai dasar teori yang mendukung pengerjaan tugas akhir.
3. Bab III. Perancangan Perangkat Lunak
   Bab perancangan perangkat lunak berisi perancangan data dan perancangan proses.
4. Bab IV. Implementasi
   Bab implementasi berisi implementasi dari desain dan perancangan perangkat lunak yang telah dijelaskan dalam bab III menggunakan Matlab R2015a.
5. Bab V. Uji Coba dan Evaluasi
   Bab ini berisi hasil uji coba perangkat lunak yang telah diimplementasikan berdasarkan skenario uji coba yang telah

dibuat. Untuk setiap uji coba dilakukan analisa dan evaluasi atas hasil yang didapatkan.

6. Bab VI. Kesimpulan dan Saran
   Bab kesimpulan dan saran berisi kesimpulan yang diambil dari tugas akhir ini beserta saran-saran yang dapat digunakan sebagai bahan pertimbangan untuk pengembangan selanjutnya.

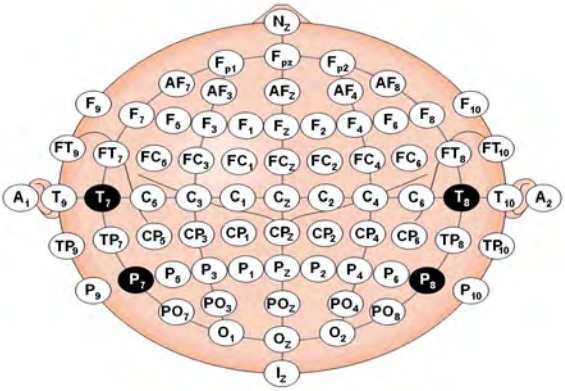
(Halaman ini sengaja dikosongkan)

BAB II
DASAR TEORI

Pada bab ini membahas dasar teori yang mendukung penyusunan tugas akhir ini. Dasar teori yang akan dijelaskan terbagi menjadi beberapa subbab yaitu *electroencephalography* (EEG), siklus tidur, *Butterworth bandpass filtering*, *autoregressive model*, *k-fold cross validation, support vector machine,* dan *confusion matrix*. Diharapkan penjelasan pada bab ini dapat memberikan gambaran secara umum mengenai perangkat lunak.

2.1 Electroencephalography (EEG)

Electroencephalography (EEG) adalah suatu metode untuk merekam aktivitas elektrik di sepanjang kulit kepala dengan mengukur fluktuasi tegangan yang dihasilkan arus ion di dalam neuron otak [2]. Sinyal EEG adalah rekaman sinyal elektronik otak selama periode waktu tertentu yang berasal dari perangkat yang disebut *Brain Computer Interface* (BCI). Sinyal-sinyal listrik yang dihasilkan oleh otak akan ditangkap oleh *channel* (berupa elektroda) pada BCI. Pemasangan elektroda-elektroda pada BCI ditunjukkan pada Gambar 2.1.



Sumber gambar : www.bci2000.org

Gambar 2.1 Titik-Titik Pemasangan Elektroda Alat Rekam EEG.

Pengukuran sinyal EEG dilakukan dengan cara meletakkan elektroda-elektroda pada kulit kepala (*scalp*). Setiap elektroda dilambangkan huruf dan angka. Huruf menandakan area dari kepala pada elektrode, semisal F ialah *Frontal Lobe* dan T ialah *Temporal lobe*. Angka genap menandakan bagian kanan dari otak dan angka ganjil menandakan bagian kiri otak. Contoh pemasangan elektroda 64 *channel* terdapat pada Gambar 2.1 dan contoh hasil perekaman EEG diditunjukkan pada Gambar 2.2.

Sumber gambar : www.neuronetrix.com

Gambar 2.2 Contoh Hasil Perekaman Sinyal EEG.

Berdasarkan frekuensi, amplitudo tegangan, dan kondisi objek, sinyal EEG dapat dibagi menjadi 4 gelombang, yaitu gelombang delta (kurang dari 4 Hz), theta (4 – 7 Hz), alpha (8 – 12 Hz), dan beta (13 – 49 Hz).

Umumnya sinyal EEG digunakan untuk mendeteksi penyakit epilepsi, mendeteksi orang dalam keadaan tidur dan bangun, mendeteksi pecandu alkohol, mendeteksi parkinson.. Hasil pengukuran dipengaruhi oleh beberapa variabel, seperti kondisi mental, aktivitas pada saat pengukuran, serta faktor stimulus eksternal yang diterima pada saat pengukuran.

2.2 Siklus Tidur

Tidur adalah cara alami manusia untuk mengembalikan energi, meregenerasi sel-sel yang rusak, dan menyembuhkan bagian tubuh yang luka. Terdapat dua fase utama tidur, yaitu *Non-Rapid Eye Movement* (NREM) dan *Rapid Eye Movement* (REM).

a. Fase tidur NREM

Non-Rapid Eye Movement (NREM) adalah siklus tidur dimana tidak terjadi pergerakan bola mata yang cepat. Siklus NREM ini hampir terjadi 80% dari siklus tidur orang normal. Siklus ini terbagi menjadi 4 tahap, yaitu :

1. Tahap 1

   Tahap ini dimulai saat kita tertidur dan berlangsung dalam waktu yang sangat singkat (sekitar 5 menit). Mata bergerak sangat lambat di bawah kelopak, aktivitas otak menurun, dan pada tahap ini kita sangat mudah terbangun. Banyak orang yang merasakan seperti sensasi "terjatuh" pada tahap ini. Sensasi tersebut menyebabkan kontraksi otot secara tiba-tiba (disebut hypnic myoclonia).

2. Tahap 2

   Tahap ini bisa dikatakan sebagai tahap awal kita benar-benar tidur, dan berlangsung antara 10-30 menit. Pada tahap ini otot tubuh menjadi sangat rileks, aktivitas otak lebih melambat, gerakan mata terhenti, gerakan mata terhenti, detak jantung melambat, dan suhu tubuh menurun.

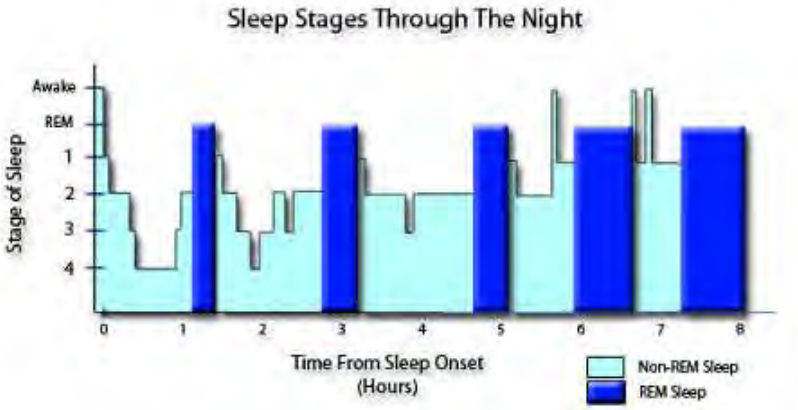
3. Tahap 3 dan 4

   Tahap 3 dan 4 merupakan tahap paling dalam dari tidur NREM. Sangat sulit terbangun pada tahap ini, dan jika terbangun kita akan mengalami disorientasi (kekacauan) serta membutuhkan penyesuaian selama beberapa menit. Pada bagian terdalam dari tahap ini, aktivitas otak sangat lambat, dan aliran darah lebih banyak diarahkan ke otot, mengisi energi fisik tubuh.

   Selama tahap tidur lelap (*deep sleep*) pada fase NREM, tubuh akan meregenerasi dan memperbaiki sel-sel tubuh, serta memperkuat sistem imunitas.

b. Fase tidur REM

Fase *Rapid Eye Movement* (REM) biasanya terjadi 70-90 menit setelah kita tertidur. Fase ini lebih dalam dari NREM. Selama fase ini, biasanya mata bergerak-gerak atau berkedut (itulah mengapa fase ini disebut *Rapid Eye Movement*) dan napas menjadi lebih tidak teratur, aktivitas otak dan ritme detak jantung juga meningkat. Umumnya mimpi terjadi pada fase ini. Namun otak melumpuhkan otot-otot tubuh, khususnya tangan dan kaki, sehingga kita tidak ikut bergerak saat bermimpi.



Sumber gambar : www.end-your-sleep-deptrivation.com

Gambar 2.3 Siklus NREM dan REM Saat Tidur.

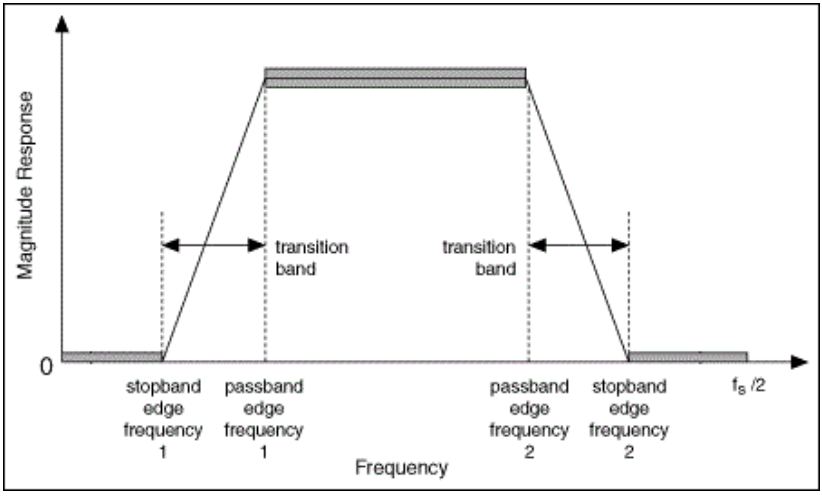
Gambar 2.3 menunjukkan selama tidur, seseorang biasanya melewati setidaknya 3 tahapan dalam NREM sebelum masuk ke fase REM. Siklus antara dua fase ini akan terus berulang selama tidur, yang masing-masingnya membutuhkan waktu antara 1-2 jam. Dan siklus ini dapat berulang sekitar 3-4 kali dalam satu malam.

Karakteristik dan pola rekaman EEG yang berkaitan dengan tahap bangun dan berbagai variasi tahapan tidur adalah *alpha band* (8-12 Hz dengan 20-60 mikroVolt Amplitudo) pada tahapan bangun, NREM1, dan REM; *beta band* (13-49 Hz dengan

2-20 mikroVolt Amplitudo) pada tahapan bangun, *theta band* (4-7 Hz dengan 50-75 mikroVolt Amplitudo) pada tahapan NREM1, NREM2, NREM3, NREM4, REM.

2.3 Butterworth Bandpass Filtering

Bandpass filter adalah filter yang hanya melewatkan sinyal-sinyal yang frekuensinya tercantum dalam pita frekuensi atau *pass band* tertentu. Frekuensi dari sinyal yang berada di bawah maupun di atas pita frekuensi tidak dapat dilewatkan oleh rangkaian *bandpass filter* [5]. Gambar 2.4 menunjukkan respon frekuensi dari *bandpass filter*.



Sumber gambar : www.zone.ni.com

Gambar 2.4 Grafik Respon Frekuensi *Bandpass Filter.*

Butterworth filter merupakan jenis *filter* pemrosesan sinyal yang didesain untuk memiliki respon frekuensi sedatar (rata) mungkin dalam *passband*. *Filter Butterworth* didefinisikan melalui persamaan *magnitude function* $H(\omega)$ (2.1).

$$|H(\omega)|^2 = \frac{1}{1+\omega^{2n}} \tag{2.1}$$

dimana ω adalah frekuensi angular dalam radian per detik, dan n adalah *order filter*.

Berikut adalah algoritma *Butterworth bandpass filtering:*

1. Hitung *zeros, poles,* dan *gain Butterworth analog lowpass filter prototype.*
   - *zeros* = []
   - *poles* = $exp^{i(\frac{\pi(2k-1)}{2n}+\frac{\pi}{2})}$; $n = order\ filter; k = 1,2,\dots,n$
   - *gain* = $\prod_{j=1}^{order} poles(j)$
2. Konversikan *zeros, poles,* dan *gain* ke dalam bentuk *state-space*.
3. Konversikan *lowpass filter* ke dalam *bandpass filter* menggunakan *state-space transformation* sesuai dengan frekuensi pembatas yang diinginkan.

Dimana zeros, poles adalah parameter yang menentukan kestabilan suatu sistem *filtering*. Sedangkan *gain* merupakan parameter yang menentukan besarnya *magnitude response.*

2.4 Autoregressive Model

Autoregressive (AR) *Model* adalah model yang menggambarkan bahwa variabel dependen dipengaruhi oleh variabel dependen itu sendiri pada periode-periode dan waktu-waktu sebelumnya [6]. Secara umum, *autoregressive model* memiliki bentuk persamaan (2.2) atau (2.3)

$$y(t) + \alpha_1 y(t-1) + \alpha_2 y(t-2) + \cdots + \alpha_p y(t-p) = e(t) \tag{2.2}$$

atau

$$y(t) = -\sum_{i=1}^{p} \alpha_i y(t-i) + e(t)\,, \text{untuk } t = 1,2,3,\dots,N \tag{2.3}$$

dimana *N* adalah panjang data pada sinyal *y*

p adalah order *autoregressive model*

α adalah parameter atau koefisien autoregressive model

y(t) adalah nilai sekarang
y(t – i) adalah nilai sebelumnya
e(t) adalah *white gaussian noise with zero mean and variance*

Pada tugas akhir ini digunakan metode *Yule-Walker* dan metode *Burg* untuk mengestimasi nilai parameter atau koefisien *autoregressive model*.

2.4.1 Metode *Yule-Walker*

Persamaan (2.4) adalah persamaan *Yule-Walker* untuk mendapatkan nilai parameter α (koefisien AR model) [7] :

$$R_{xx}(k) + \sum_{i=1}^{p} \alpha_i R_{xx}(k-i) = 0; \;\; k = 1,2,\dots,order \qquad (2.4)$$

dengan *error* model didefinisikan dengan rumus (2.5)

$$e = \; R_{xx}(0) + \sum_{i=1}^{p} \alpha_i R_{xx}(k-i) \qquad (2.5)$$

dimana R_{xx} adalah *autocorrelation coefficient*, dapat dihitung dengan rumus (2.6)

$$R_{xx}(k) = \frac{1}{N} \sum_{t=k+1}^{N} y(t) y^*(t-k) \qquad (2.6)$$

y^* *merupakan kompleks konjugasi dari* y.

Untuk mempermudah perhitungan, persamaan (2.4) dan (2.5) dikombinasikan kemudian diubah ke dalam persamaan matriks menjadi:

$$\begin{bmatrix} R_{xx}(0) & R^*_{xx}(1) & \dots & R^*_{xx}(p-1) & R^*_{xx}(p) \\ R_{xx}(1) & R_{xx}(0) & & R^*_{xx}(p-2) & R^*_{xx}(p-1) \\ \vdots & \vdots & & \vdots & \vdots \\ R_{xx}(p-1) & R_{xx}(p-2) & & R_{xx}(0) & R^*_{xx}(1) \\ R_{xx}(p) & R_{xx}(p-1) & \dots & R_{xx}(1) & R_{xx}(0) \end{bmatrix} \begin{bmatrix} 1 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \vdots \\ \alpha_p \end{bmatrix} = \begin{bmatrix} e \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

Perhitungan persamaan matriks di atas dapat diselesaikan menggunakan algoritma *Levinson-Durbin recursion*. Penyelesaian *Levinson-Durbin recursion* dilakukan dengan menambahkan parameter γ dan konstanta Γ yang didefinisikan melalui persamaan (2.7) dan (2.8).

$$\gamma = R_{xx}(j+1) + \sum_{i=2}^{j} \alpha_i \, R_{xx}(j-i+2) \tag{2.7}$$

$$\Gamma = \frac{-\gamma}{error} \tag{2.8}$$

Berikut adalah *pseudocode* algoritma *Levinson-Durbin recursion*:

1. Inisialisasi
   a) $\alpha_1 = 1$
   b) $error = R_{xx}(0)$
2. For j = 1 to *order*
   a) Hitung parameter γ
   b) Hitung konstanta Γ
   c) for i = 2 to j
      $\alpha_i = \alpha_i + \Gamma \alpha^*_{(j-i+2)}$
   d) $\alpha_{j+1} = \Gamma$
   e) $error = error(1 - |\Gamma|^2)$

Koefisien atau parameter *autoregressive model* merupakan hasil akhir variabel $\alpha_1, \alpha_2, \dots, \alpha_{order+1}$, dari algoritma *Levinson-Durbin recursion* [7].

2.4.2 Metode *Burg*

Pada tahun 1975, Burg mengembangkan suatu metode untuk mengestimasi parameter AR Model yang didasarkan pada prediksi linier *forward* dan *backward* [8]. Misalkan diketahui data signal y(t) dengan *t = 1,2,…,N*. Maka kita bisa menghitung prediksi linier *forward* dan *backward* menggunakan persamaan (2.9) dan (2.10) secara berurutan

$$y'(t) = -\sum_{i=1}^{p} \alpha_{p,i} y(t-i) \tag{2.9}$$

$$y'(t-p) = -\sum_{i=1}^{p} \alpha^*_{p,i} y(t-i) \tag{2.10}$$

dimana α dan α^* secara berurutan adalah koefisien prediksi linier *forward* dan *backward* dan p adalah order AR model. α^* didefinisikan sebagai kompleks konjugasi dari α. Sedangkan prediksi *error forward* dan error *backward* secara berututan dapat dihitung dengan rumus (2.11) dan (2.12)

$$ef_p(t) = y(t) - y'(t) = y(t) + \sum_{i=1}^{p} \alpha_{p,i} y(t-i) \tag{2.11}$$

$$eb_p(t) = y(t-p) + \sum_{i=1}^{p} \alpha^*_{p,i} y(t-p+i) \quad untuk\ t = p+1, \dots, N \tag{2.12}$$

$ef_p(t)$ dan $ef_b(t)$ berurutan adalah prediksi *error forward* dan *backward*. $ef_p(t)$ dan $ef_b(t)$ bergantung pada nilai *p+1*, sehingga nilai *p* harus lebih kecil dari *N*. ef_p dan ef_b dapat diminimalisasi

dengan menambahkan suatu *reflection coefficient* k_p yang memenuhi persamaan (2.13)

$$k_p = \frac{-2\sum_{t=p+1}^{N} ef_{p-1}(t) + eb^*{}_{p-1}(t-1)}{\sum_{t=p+1}^{N} \left| \left| ef_{p-1}(t) \right|^2 + \left| eb_{p-1}(t-1) \right|^2 \right|} \tag{2.13}$$

Berikut adalah algoritma *Burg's method* untuk mengestimasi parameter AR model :

1. Inisialisasi $ef_0(t) = eb_0(t) = y(t)$.
2. *For p=1 to order AR*
   a. Hitung *reflection coefficient* k_p menggunakan rumus (2.13)
   b. Perbaharui nilai $ef(t)$, $eb(t)$ dengan rumus (2.14) dan (2.15)

$$ef_p(t) = ef_{p-1}(t) + k_p eb_{p-1}(t-1) \tag{2.14}$$
$$eb_p(t) = eb_{p-1}(t-1) + k^*{}_p ef_{p-1}(t) \tag{2.15}$$

   c. Hitung $\alpha_{p,i}$ untuk *i = 1,...,p* dengan rumus (2.16) dan (2.17):

$$\alpha_{p,i} = \alpha_{p-1,i} + k_p \alpha^*{}_{p-1,p-i}, untuk\ i = 1, \dots, p-1 \tag{2.16}$$
$$\alpha_{p,i} = k_p, untuk\ i = p \tag{2.17}$$

3. $\alpha_{p,1}, \dots, \alpha_{p,p}$ adalah hasil estimasi koefisien AR model.

2.5 *K-Fold Cross Validation*

Dalam *k-fold cross validation*, dataset asli dibagi secara acak menjadi *k* subdataset berukuran sama. Satu subdataset dari *k* subdataset digunakan sebagai data uji dan *k-1* subdataset lainnya digunakan sebagai data latih. Proses *cross-validation* kemudian diulang sebanyak *k* kali, dengan masing-masing subdataset *k* digunakan tepat sekali sebagai data uji [9]. Gambar 2.5 merupakan visualisasi *10-fold cross validation*.

Sumber gambar : www.sebastianraschka.com

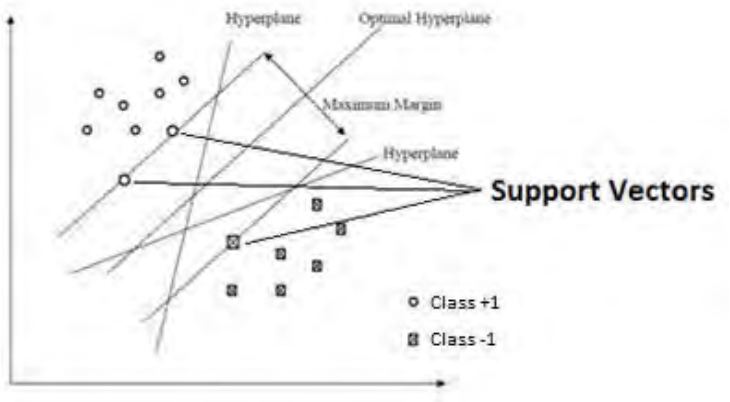
Gambar 2. 5 Visualisasi *10-Fold Cross Validation*

2.6 *Support Vector Machine*

Support Vector Machine (SVM) dikembangkan oleh Oser, Guyon, Vapnik dan pertama kali dipresentasikan pada tahun 1992. Prinsip dasar SVM adalah *linear classifier*, dan selanjutnya dikembangkan agar dapat bekerja pada *problem non-linear* dengan memasukkan konsep *kernel trick* pada ruang kerja berdimensi tinggi.

Konsep dasar SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua kelas pada *input space*. Gambar 2.5 merupakan beberapa contoh dari kemungkinan *hyperplane* yang digunakan untuk memisahkan kelas satu dengan lainnya [10].

Hyperplane pemisah terbaik antara kedua kelas dapat ditemukan dengan mengukur *margin hyperplane* dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperplane* dengan data terdekat dari masing-masing kelas. Data yang paling dekat dengan *hyperplane* disebut sebagai *support vectors*.



Sumber gambar : www.sine.ni.com

Gambar 2.6 Contoh Alternatif *Hyperplane*.

Diberikan data masukan $x_i \in \Re^d$ dan masing-masing kelas dinotasikan $y_i \in \{-1, +1\}$ untuk $i = 1,2,3, \ldots, n$ dimana n adalah banyaknya data. Fungsi *hyperplane* dibuat dengan persamaan (2.18)

$$w \,.\, x + b = 0 \tag{2.18}$$

Dimana w adalah vektor normal *hyperplane* tersebut, dan b adalah bias. Data x_i yang termasuk kelas -1 dapat dirumuskan sebagai data yang memenuhi pertidaksamaan (2.19)

$$w \,.\, x_i + b \leq -1 \tag{2.19}$$

sedangkan data x_i yang termasuk kelas +1 dirumuskan sebagai data yang memenuhi pertidaksamaan (2.20)

$$w \,.\, x_i + b \geq +1 \tag{2.20}$$

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara *hyperplane* dan titik terdekatnya dengan rumus (2.21)

$$\frac{1}{||w||^2} \tag{2.21}$$

Memaksimalkan persamaan (2.21) dapat diselesaikan dengan mencari titik minimal dari fungsi $\tau(w)$ yang didefinisikan melalui persamaan (2.22) dengan memperhatikan *constraint* pada persamaan (2.23).

$$\min \tau(w) = \min\left(\frac{1}{2}||w||^2\right) \tag{2.22}$$

$$y_i(x_i.w + b) - 1 \geq 0, \forall i \tag{2.23}$$

Masalah ini dapat dipecahkan dengan berbagai teknik komputasi di antaranya *Lagrange Multiplier* melalui persamaan (2.24).

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i(y_i\big((x_i.w + b) - 1\big)) \tag{2.24}$$

dimana α adalah *Lagrange multipliers*, yang bernilai nol atau positif ($\alpha_i \geq 0$). Nilai optimal dari persamaan (2.24) dapat dihitung dengan meminimalkan *L* terhadap *w* dan *b*, dan memaksimalkan *L* terhadap α_i. Dengan memperhatikan bahwa pada titik optimal *L=0*, persamaan (2.22) dapat dimodifikasi sebagai maksimalisasi *problem* yang hanya mengandung α_i saja, sebagaimana persamaan (2.25) dan (2.26)

maximize:

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j x_i x_j \tag{2.25}$$

Subject to:

$$\alpha_i \geq 0\ (i = 1,2, \dots, n) \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.26}$$

Hasil dari perhitungan ini diperoleh α_i yang kebanyakan bernilai positif. Data yang berkorelasi dengan α_i yang positif inilah yang disebut sebagai *support vector*.

2.5.1 *Soft Margin*

Pada umumnya, dua buah kelas pada *input space* tidak dapat dipisahkan secara sempurna. Hal ini menyebabkan *constraint* pada persamaan (2.23) tidak dapat terpenuhi, sehingga optimasi tidak dapat dilakukan. Untuk mengatasi masalah ini, SVM dirumuskan ulang dengan memperkenalkan teknik *soft margin*. Dalam *soft margin*, persamaan (2.23) dimodifikasi dengan memasukkan variabel *slack* ξ_i. Penambahan variabel *slack* ξ_i berfungsi untuk menghitung tingkat kesalahan klasifikasi, dimana $\xi_i > 0$. Sehingga persamaan (2.23) dapat diubah menjadi persamaan (2.27).

$$y_i(x_i.w + b) \geq 1 - \xi_i, \qquad \forall i \tag{2.27}$$

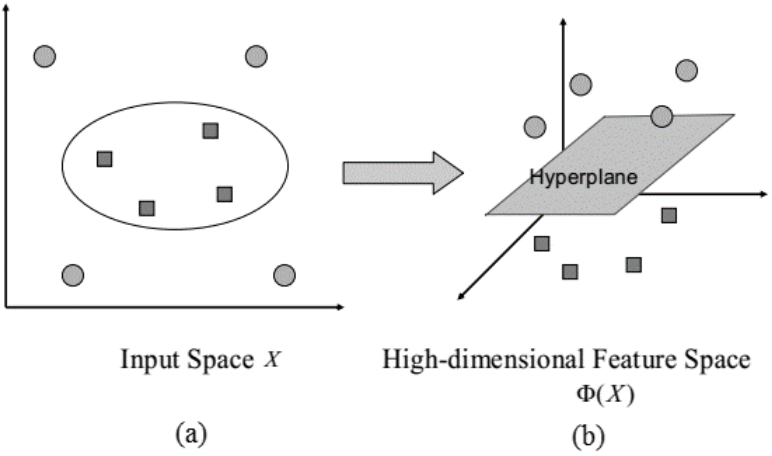
Dengan demikian persamaan (2.21) diubah menjadi persamaan (2.28)

$$\min \tau(w, \xi) = \min\left(\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i\right) \tag{2.28}$$

Parameter *C* adalah konstanta nilai pinalti dari kesalahan klasifikasi. Parameter C dipilih untuk mengontrol *tradeoff* antara *margin* dan *error* klasifikasi ξ. Nilai parameter C mempengaruhi besar kecilnya *margin hyperplane.* Semakin besar nilai C, maka *margin hyperplane* akan semakin kecil. Sebaliknya, nilai C yang besar mengakibatkan semakin kecilnya *margin hyperplane*. [10].

2.5.2 *Kernel Trick*

Pada umumnya masalah klasifikasi dalam dunia nyata jarang yang bersifat *linear separable*, kebanyakan bersifat *non-linear.* Untuk menyelesaikan masalah *non-linear*, *support vector machine* (SVM) dimodifikasi dengan memasukkan fungsi *kernel* [10]. Penggnaan fungsi *kernel* bertujuan untuk memetakan data yang bersifat *non-linear* menjadi data dengan dimensi yang lebih tinggi.



Gambar 2.7 Fungsi Φ(x) Memetakan Data ke Ruang Vektor Lebih Tinggi sehingga Kedua Kelas Dapat Dipisahkan Secara Linier oleh Sebuah *Hyperplane* [10].

Dalam *non-linear* SVM, pertama-tama data x dipetakan oleh fungsi $\Phi(x)$ ke ruang vektor yang berdimensi lebih tinggi. Pada ruang vektor yang baru ini, *hyperplane* yang memisahkan kedua kelas tersebut dapat dikonstruksikan. Pada Gambar 2.6 (a) diperlihatkan data pada kelas +1 yang direpresentasikan dalam bentuk lingkaran dan kelas -1 yang direpresentasikan dalam bentuk persegi berada pada *input space* berdimensi dua tidak dapat dipisahkan secara linier. Selanjutnya Gambar 2.6 (b) menunjukkan bahwa fungsi Φ(x) memetakan tiap data pada *input space* tersebut ke ruang vektor baru yang berdimensi lebih tinggi (dimensi 3), dimana kedua kelas dapat dipisahkan secara linier oleh sebuah *hyperplane*. Notasi matematika dari pemetaan ini ditunjukkan pada persamaan (2.29).

$$\Phi(\mathrm{x}) : \Re^d \rightarrow \Re^q \; dengan \; d < q \qquad (2.29)$$

Persamaan (2.29) berarti bahwa fungsi Φ(x) memetakan data x yang memiliki dimensi d menjadi data yang memiliki dimensi lebih

tinggi. Pemetaan ini dilakukan dengan menjaga topologi data, dalam artian dua data yang berjarak dekat pada *input space* akan berjarak dekat juga pada *feature space* begitupun data yang berjarak jauh pada *input space* akan berjarak jauh pada *featur space*.

Karena umumnya fungsi transformasi $\Phi(x)$ ini tidak diketahui, dan sangat sulit untuk dipahami, maka fungsi transformasi $\Phi(x)$ dapat digantikan dengan fungsi *kernel* yang mendefinisikan secara implisit fungsi transformasi $\Phi(x)$. Hal ini disebut sebagai *kernel trick* yang dirumuskan berdasarkan persamaan (2.30)

$$\mathrm{K}(x_i, x_j) = \Phi(x_i).\Phi(x_j) \tag{2.30}$$

Kernel trick memberikan berbagai kemudahan, karena dalam proses pembelajaran SVM, untuk menentukan *support vector*, kita hanya cukup mengtahui fungsi *kernel* yang dipakai, dan tidak perlu mengetahui wujud dari fungsi *non-linear*. Jenis-jenis *kernel* yang umum digunakan dalam SVM dapat dilihat pada Tabel 2.1.

Tabel 2.1 Jenis *Kernel* yang Umum Digunakan dalam SVM

Jenis kernel	Definisi
Polynomial	$\mathrm{K}(x_i, x_j) = (x_i.x_j + 1)^p$
Gaussian (RBF)	$\mathrm{K}(x_i, x_j) = \exp(\frac{\lvert\lvert x_i - x_j \rvert\rvert^2}{2\gamma^2})$

Selanjutnya hasil klasifikasi dari data x diperoleh dari persamaan (2.31), (2.32), dan (2.33).

$$f(\Phi(x)) = w.\Phi(x) + b \tag{2.31}$$

$$f(\Phi(x)) = \sum_{i=1, x_i \in SV}^{n} \alpha_i y_i \Phi(x).\Phi(x_i) + b \tag{2.32}$$

$$f(\Phi(x)) = \sum_{i=1, x_i \in SV}^{n} \alpha_i y_i \, K(x, x_i) + b \tag{2.33}$$

SV pada persamaan (2.32) dan (2.33) dimaksudkan dengan subset dari data *training* yang terpilih sebagai *support vectors*, dengan kata lain data x_i yang berkorespondensi pada $\alpha_i \geq 0$.

2.6 *Confusion Matrix*

Confusion matrix adalah suatu metode yang biasa digunakan untuk melakukann perhitungan performa suatu alogritma pada konsep data mining. *Confusion matrix* memiliki informasi hasil prediksi dan aktual pada data yang telah diklasifikasi. Tabel 2.2 menunjukkan *confusion matrix* untuk dataset yang memiliki dua kelas.

Tabel 2.2 *Confusion Matrix* Dua Kelas

		Prediksi	
		1	0
Aktual	1	TP	FN
	0	FP	TN

Nilai performa yang bisa dihitung menggunakan *confusion matrix* antara lain: akurasi, sensitivitas, presisi. Akurasi adalah hasil bagi dari jumlah prediksi yang terklasisfikasi secara benar dibagi total data yang diklasifikasi seperti pada persamaan (2.34).

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.34}$$

Presisi adalah perbandingan dari jumlah data TP dengan total data TP dan FP seperti pada persamaan (2.35).

$$Presisi = \frac{TP}{TP + FP} \tag{2.35}$$

Sensitivitas atau *recall* adalah perbandingan dari jumlah data TP dengan total data TP dan FN seperti pada persamaan (2.36).

$$Sensitivitas\ atau\ Recall = \frac{TP}{TP + FN} \tag{2.36}$$

(Halaman ini sengaja dikosongkan)

BAB III
PERANCANGAN PERANGKAT LUNAK

Pada bab ini berisi penjelasan tentang perancangan sistem perangkat lunak untuk mencapai tujuan dari tugas akhir. Perancangan yang diuraikan meliputi peracangan data dan perancangan proses. Perancangan data adalah proses perancangan pemilihan data yang akan digunakan dalam proses latih dan proses uji. Penjelasan perancangan proses terbagi menjadi 3 yaitu tahap preprocessing, tahap ekstraksi fitur menggunakan metode *autoregressive model*, serta tahap klasifikasi menggunakan metode *support vector machine*.

3.1 Perancangan Data

Pada sub bab ini akan dijelaskan mengenai perancangan data yang dibutuhkan untuk membangun perangkat lunak. Perancangan data meliputi data masukan, data proses berupa data-data yang dibutuhkan dan dihasilkan selama menjalankan sub proses eksekusi perangkat lunak, dan data keluaran.

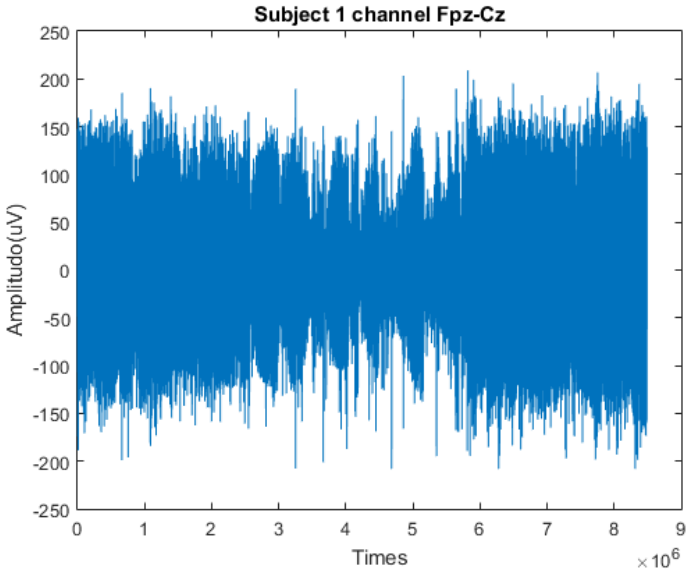
3.1.1 Data masukan

Data masukan adalah data yang digunakan sebagai masukan awal dari sistem. Dataset yang digunakan dalam tugas akhir ini adalah data sinyal otak (EEG) yang diunduh dari www.physionet.org/cgi-bin/atm/ATM (*Sleep-EDF Database, v1 [deprecated, use sleep-edfx]*). Dataset terdiri 8 subjek (yang memiliki durasi rekaman berbeda-beda), masing-masing subjek terdiri dari 2 tipe yaitu data hasil rekaman EEG dan data *hypnogram* (grafik yang menunjukkan tahapan tidur sebagai fungsi waktu). Data hynogram berisi *ground truth class* yang diperlukan untuk proses latih pada klasifikasi, keterangan kelas tersebut antara lain:

- 0 untuk kelas bangun

- 1 untuk kelas tidur NREM1
- 2 untuk kelas tidur NREM2
- 3 untuk kelas tidur NREM3
- 4 untuk kelas tidur NREM4
- 5 untuk kelas tidur REM

Data hasil rekaman EEG dengan format EDF (European Define Format) terlebih dahulu diubah ke dalam format file text (.txt) menggunakan aplikasi *EDF Browser*. Gambar 3.1 merupakan contoh dari data EEG subjek 1 *channel Fpz-Cz* yang berdurasi 23 jam 35 menit 30 detik. *Channel Fpz-Cz* dapat didefiniskan sebagai perbedaan tegangan antara elektrode *Fpz* dan elektrode *Cz*.



Gambar 3.1 Data EEG Subjek1.

Setelah diubah ke dalam format file text (.txt) data dibagi menjadi beberapa file dengan durasi masing-masing 30 detik. Data hasil bagi tersebut berisi nilai yg didapat dari *channel Fpz-Cz* pada

kolom 1, *Pz-Oz* pada kolom 2, dan kelas yang berasal dari file *hypnogram* pada kolom 3. Karena frekuensi sampel dari dataset 100 Hz, maka setiap file yang sudah dibagi menjadi 30 detik terdapat 3000 nilai. Gambar 3.2 merupakan isi dari data yang sudah dibagi menjadi 30 detik.

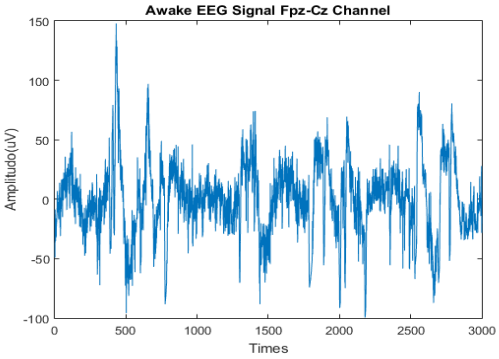
data

3000x3 double

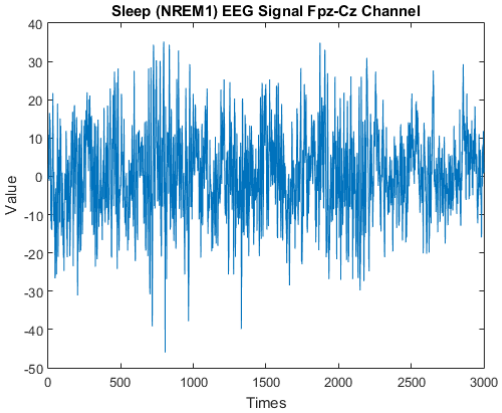
	1	2	3
1	-5.2553	1.8373	5
2	-4.9499	8.0205	5
3	-11.4656	8.0205	5
4	-8.5132	7.9224	5
5	-12.6873	5.8613	5
6	-7.1897	5.2724	5
7	-12.2801	3.6039	5
8	-4.3391	2.7206	5
9	-11.6693	2.1317	5
10	-14.3163	-1.9965	5
11	-4.8481	-3.4687	5
12	-6.8843	-2.0947	5

Gambar 3.2 Isi Data yang Sudah Dibagi menjadi 30 detik

Pada tugas akhir ini, data masukan yang digunakan adalah data sinyal EEG pada *channel Fpz-Cz.*



Gambar 3.3 Contoh Data EEG (*Fpz-Cz channel*) untuk Kelas Bangun.



Gambar 3.4 Contoh Data EEG (*Fpz-Cz channel*) untuk Kelas NREM1.

Gambar 3.3 dan 3.4 menunjukkan data EEG *channel Fpz-Cz* yang sudah dibagi dalam durasi 30 detik.

3.1.2 Data proses

Data proses merupakan data keluaran dari data masukan awal dan menjadi data masukan pada proses selanjutnya. Pada perangkat lunak ini, terdapat 3 tahapan secara garis besar, yaitu tahap *prepocessing* yang meliputi normalisasi dan *filtering*, tahap ekstraksi fitur, serta tahap klasifikasi. Sebelum dilakukan klasifikasi, data pada tiap subband dibagi menjadi data latih dan data uji menggunakan metode *10-fold cross validation*. Data latih digunakan untuk membentuk *SVM model* yang digunakan untuk mengklasifikasikan data uji. Selanjutnya, data kelas hasil klasifikasi pada setiap *subband* akan menjadi masukan pada proses penentuan kelas final menggunakan metode *majority vote.*

Data atau variabel penting hasil dari proses *preprocessing* dan yang akan digunakan pada proses ekstraksi fitur dapat dilihat pada Tabel 3.1.

Tabel 3.1 Data Hasil dari Preprocessing

No.	Nama Data	Keterangan
1	*signal*	Data EEG *channel Fpz-Cz*
2	*hypnogram*	*Ground truth class* (kelas 0 untuk bangun, kelas 1 untuk tidur)
3	*signalnorm*	Data sinyal EEG yang sudah dinormalisasi dengan *range* nilai [-1, 1]
4	*thetaband*	Data setelah dilakukan *butterworth bandpass filtering* (4-7 Hz)
5	*alphaband*	Data setelah dilakukan *butterworth bandpass filtering* (8-12 Hz)
6	*betaband*	Data setelah dilakukan *butterworth bandpass filtering* (13-49 Hz)

Data atau variabel hasil dari proses ekstraksi fitur yang akan digunakan sebagai data masukan pada proses klasifikasi dapat dilihat pada Tabel 3.2.

Tabel 3.2 Data Hasil dari Proses Ekstraksi Fitur.

No.	Nama Data	Keterangan
1	*theta_attributes*	Data hasil ekstraksi fitur theta band
2	*alpha_attributes*	Data hasil ekstraksi fitur alpha band
3	*beta_attributes*	Data hasil ekstraksi fitur beta band

Sebelum dilakukan proses klasifikasi, data hasil ekstraksi fitur dan label (*ground truth* class) data tersebut dibagi menjadi data latih dan data uji. Tabel 3.3 menunjukkan data atau variabel latih hasil *10-fold cross validation*.

Tabel 3.3 Data atau Variabel Latih Hasil 10-Fold Cross Validation.

No.	Nama Data	Keterangan
1	*theta_attributes_training*	Data latih atribut theta band
2	*alpha_attributes_training*	Data latih atribut alpha band
3	*beta_attributes_training*	Data latih atribut beta band
4	*hypno_training*	Label atau kelas data latih

Data atau variabel uji hasil *10-fold cross validation* ditunjukkan pada Tabel 3.4.

No.	Nama Data	Keterangan
1	*theta_attributes_testing*	Data uji atribut theta band
2	*alpha_attributes_ testing*	Data uji atribut alpha band
3	*beta_attributes_ testing*	Data uji atribut beta band
4	*hypno_ testing*	Label atau kelas data uji

3.1.3 Data keluaran

Perangkat lunak ini menghasilkan klasifikasi dari sinyal EEG. Hasil klasifikasi dari data uji akan dijadikan data keluaran Data keluaran dari perangkat lunak berupa angka yaitu 0 untuk kelas bangun, dan 1 untuk kelas tidur.

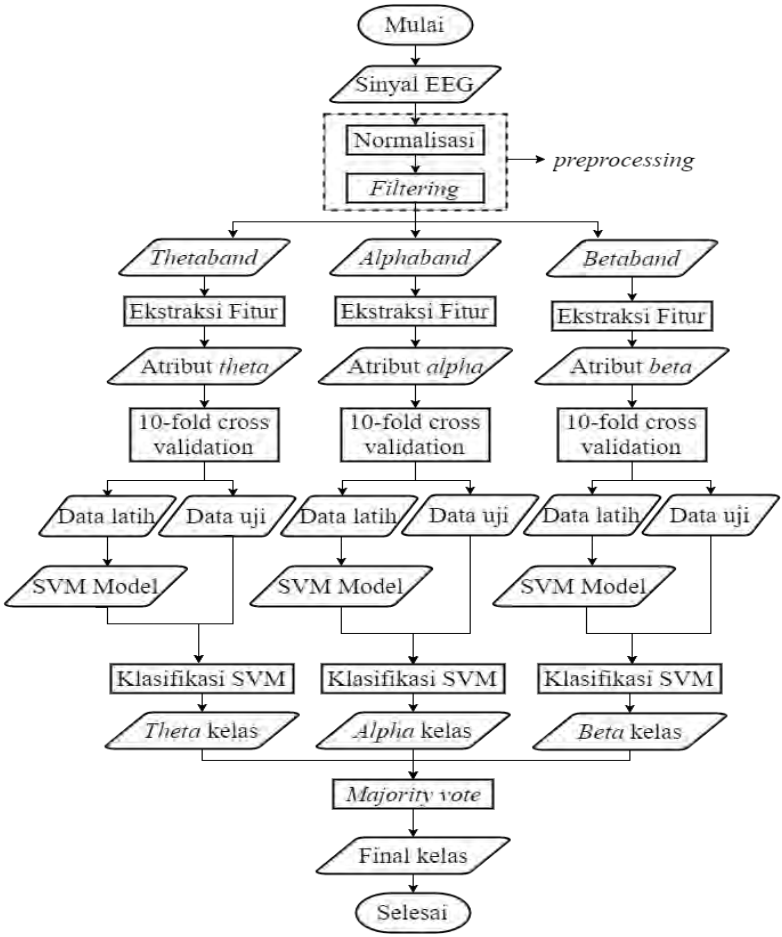
3.2 Deskripsi Umum Perangkat Lunak

Rancangan perangkat lunak pada tugas akhir ini menggunakan autoregressive model sebagai metode ekstraksi fitur dan *support vector machine* sebagai metode klasifikasi. Data hasil ekstraksi fitur akan dijadikan data latih pada *support vector machine* sehingga akan menghasilkan suatu model. Kemudian dilakukan uji coba (*testing*) pada model yang dihasilkan dengan menggunakan data uji yang telah tersedia.

Tahap pertama dari perangkat lunak ini adalah preprocessing dimana sinyal EEG sebagai data masukan awal dinormalisasikan dengan range [-1 1]. Setelah dinormalisasi, data di-*filter* menggunakan *butterworth bandpass filtering* dengan *order* 8 untuk *theta band* (4-7 Hz), *order* 10 untuk *alpha band* (8-12 Hz), *order* 11 untuk *beta band* (13-49 Hz).

Data hasil tahap *preprocessing*, akan menjadi masukan pada tahap ekstraksi fitur. Pada tugas akhir ini, metode yang digunakan untuk ekstraksi fitur adalah *autoregressive model*. Data hasil *filtering theta band, alpha band,* dan *beta band* masing-

masing dimasukkan dalam tahap ekstraksi fitur, dan keluaran dari proses ekstraksi fitur adalah *n*-koefisien dari *autoregressive model* dengan *n* adalah *order* dari autoregressive model. Sehingga keluaran dari tahap ekstraksi fitur adalah atribut dari *theta, alpha,* dan *beta*. Diagram alir perangkat lunak secara umum ditunjukkan pada Gambar 3.5.



Gambar 3.5 Diagram Alir Sistem Secara Umum.

Data atribut *theta, alpha,* dan *beta* masing-masing dilakukan klasifikasi secara terpisah. Penentuan data latih dan data uji menggunakan metode *10-fold cross validation*. Hasil klasifikasi dari setiap band (*theta, alpha, beta*) dilakukan *majority vote* untuk menentukan final kelas.

3.3 Perancangan Proses

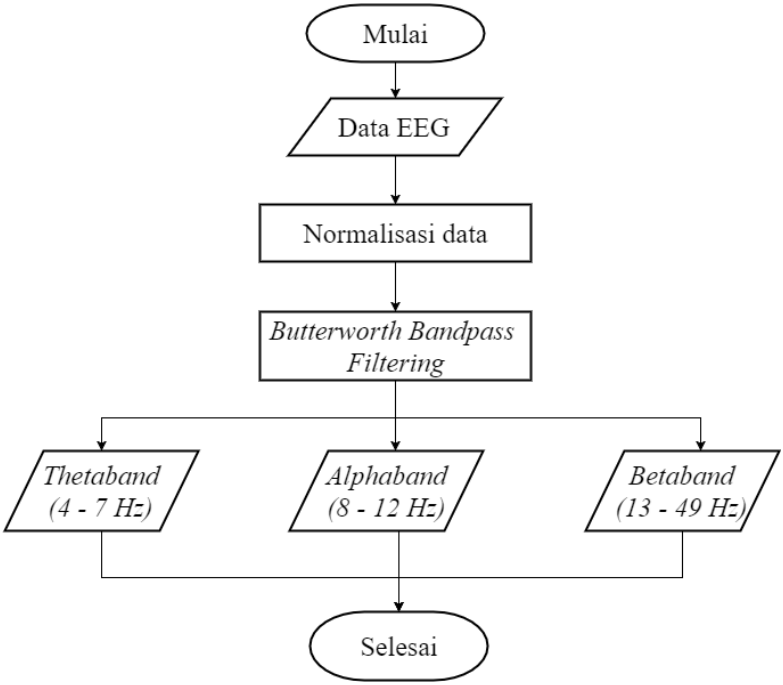
Perancangan proses merupakan tahap untuk membentuk alur dalam penerapan algoritma *autoregressive model* dan *support vector machine* yang nantinya akan diimplementasikan untuk pembentukan prangkat lunak. Perancangan proses meliputi tahap *preprocessing*, tahap ekstraksi fitur, dan tahap klasifikasi.

3.3.1 Tahap *Preprocessing*

Pada tahap *preprocessing*, diberikan data masukan sinyal EEG X_i dimana *i=1,2,…,* panjang data. Data masukan dinormalisasikan dengan *range* [-1 1]. Hal tersebut dilakukan agar data memilki skala yang sama. Normalisasi data dilakukan dengan rumus:

$$X_{new} = \frac{(X_{old} - \min(X_{old})) * 2}{\max(X_{old}) - \min(X_{old})} - 1 \qquad (3.1)$$

Setelah dilakukan normalisasi, data di-*filter* menggunakan *Butterworth bandpass filtering* untuk memisahkan data menjadi 3 sub band yaitu *theta, alpha, dan beta*. *Order* 8 dan batas frekuensi 4 - 7 Hz untuk *theta*, *order* 10 dan batas frekuensi 8 - 12 Hz untuk *alpha*, *order* 11 dan batas frekuensi 13 - 49 Hz untuk *beta*. Hasil dari proses *filtering* akan menjadi data masukan pada tahap ekstraksi fitur. Gambar 3.6 merupakan diagram alir dari tahap *preprocessing* (yang terdiri dari proses normalisasi dan *filter* data) data EEG.

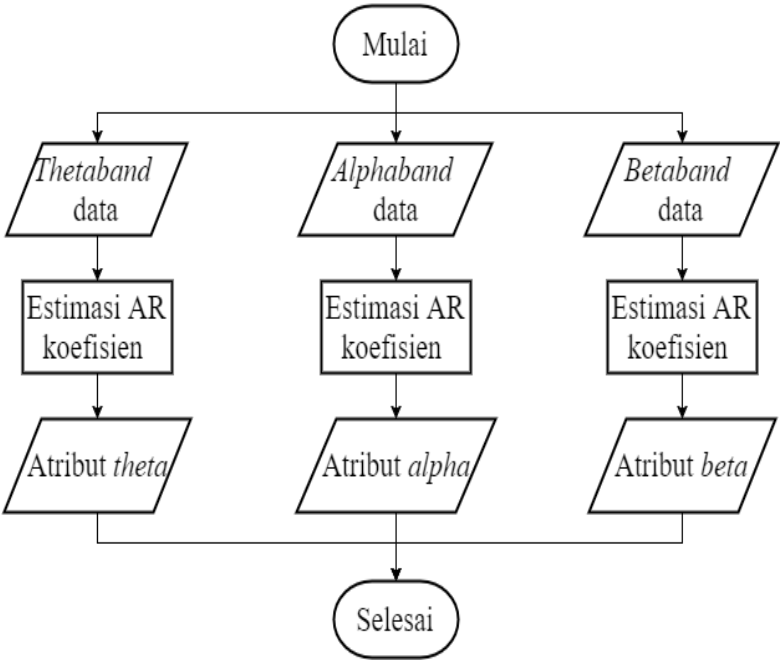


Gambar 3.6 Diagram Alir Tahap *Preprocessing* Data EEG.

3.3.2 Tahap Ekstraksi Fitur

Setelah terbagi menjadi *theta*, *alpha*, dan *beta*, pada masing-masing sub band tersebut akan dilakukan tahap ekstraksi fitur menggunakan metode *autoregressive model*.

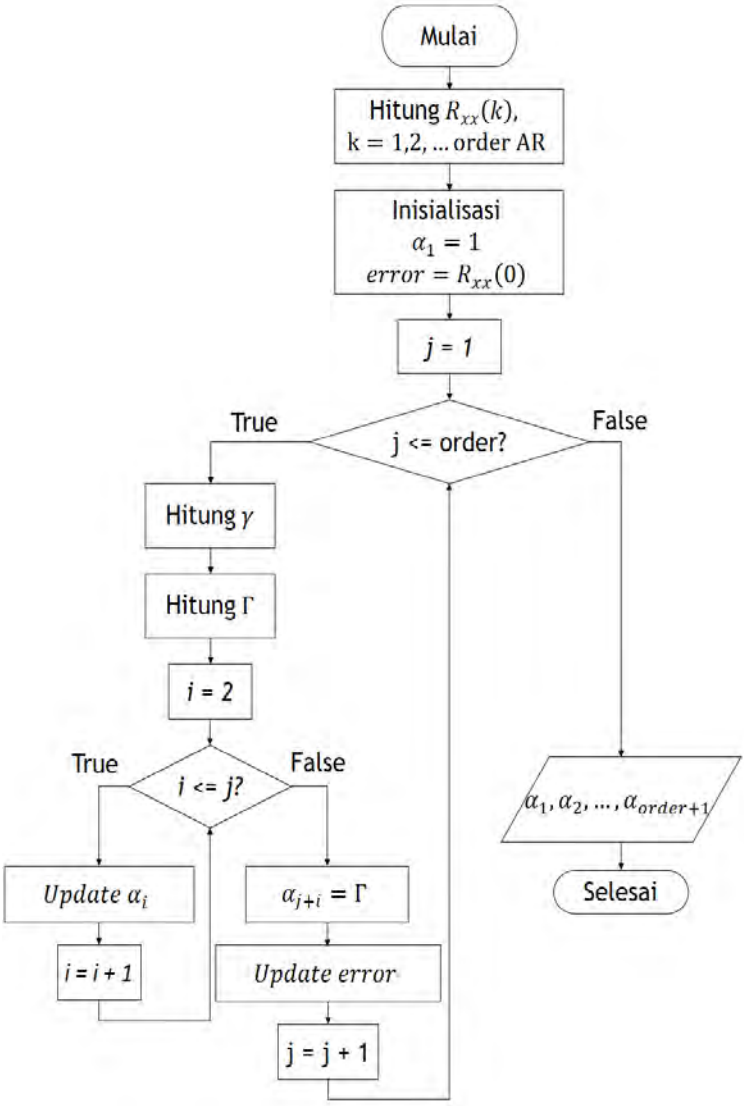
Koefisien *autoregressive model* akan menjadi fitur yang merupakan data masukan pada tahap klasifikasi. Jumlah koefisien *autoregressive model* bergantung pada nilai *order*-nya. Sehingga jumlah fitur sama dengan nilai *order* AR model. Nilai order AR model akan dijadikan salah satu skenario pengujian yang akan dijelaskan pada sub bab 5.3.1. Gambar 3.7 merupakan diagram alir dari tahap ekstraksi fitur menggunakan metode *autoregressive* (AR) *model*.



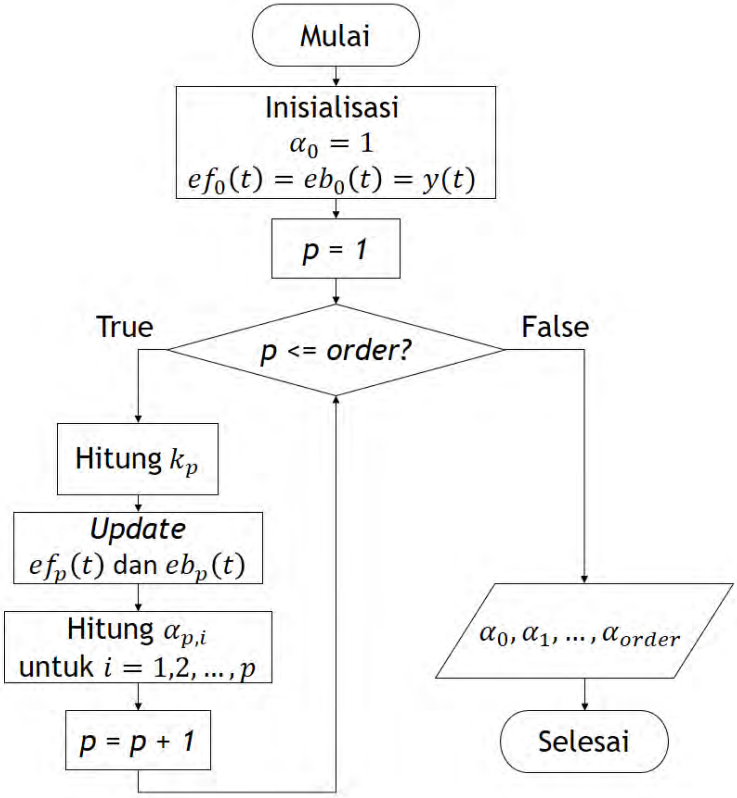
Gambar 3.7 Diagram Alir Tahap Ekstraksi Fitur.

Pada tugas akhir ini, metode yang digunakan untuk mengestimasi koefisien *autoregressive model* yaitu metode *Yule-Walker* dan metode *Burg*.

Metode *Yule-Walker* dalam mengestimasi koefisien atau parameter *autoregressive* (AR) model didasarkan pada koefisien *autocorrelation* dan *Levinson-Durbin recursion* seperti yang sudah dijelaskan pada sub bab 2.4.1. Sedangkan metode *Burg* dalam mengestimasi koefisien atau parameter *autoregressive* (AR) model didasarkan pada meminimalkan prediksi *error forward* dan prediksi *error backward*. Prediksi *error forward* dan prediksi *error backward* dapat diminimalkan dengan menambahkan koefisien *reflection* seperti yang sudah dijelaskan pada sub bab 2.4.2. Diagram alir metode *Yule-Walker* ditunjukkan pada Gambar 3.8 dan diagram alir metode *Burg* ditunjukkan pada Gambar 3.9.



Gambar 3.8 Diagram Alir Metode *Yule-Walker*

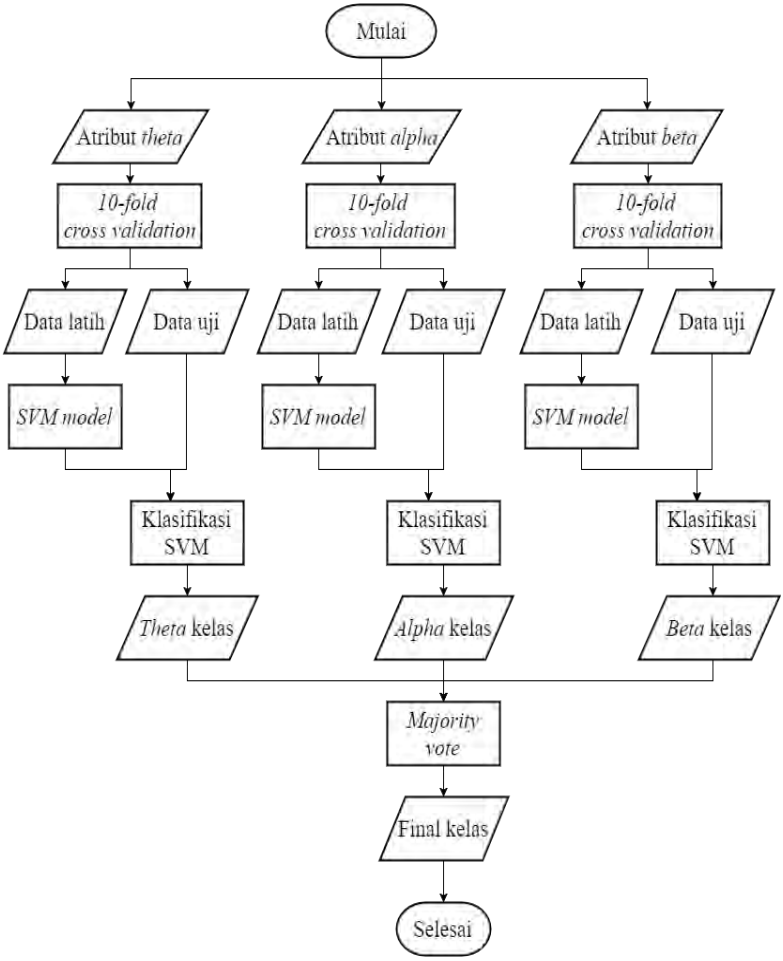


Gambar 3.9 Diagram Alir Metode *Burg*.

3.3.3 Tahap Klasifikasi

Atribut *theta*, *alpha*, dan *beta* hasil tahap ekstraksi fitur menjadi masukan pada tahap klasifikasi. Sebelum dilakukan proses klasifikasi, *theta*, *alpha*, dan *beta* masing-masing dibagi menjadi data training dan data *testing*. Data *training* adalah data yang digunakan untuk membuat model klasifikasi yang digunakan. Sedangkan data *testing* digunakan untuk proses uji coba model klasifikasi yang telah dibuat. Metode klasifikasi yang digunakan

adalah *support vector machine*. Hasil dari tahap klasifikasi adalah kelas bangun yang direpresentasikan dengan angka 0 dan kelas tidur dengan angka 1 untuk masing-masing *theta, alpha, beta*. Tahapan-tahapan dalam klasifikasi ditunjukkan pada Gambar 3.10 dalam bentuk diagram alir.



Gambar 3.10 Diagram Alir Tahap Klasifikasi.

Ketiga hasil klasifikasi akan dilakukan *majority vote* sehingga menghasilkan final kelas. Tabel 3.3 adalah contoh proses penentuan final kelas menggunakan *majority vote*.

Tabel 3.4 Contoh Penentuan Final Kelas Menggunakan *Majority Vote*.

Kelas pada *thetaband*	Kelas pada *alphaband*	Kelas pada *betaband*	Final Kelas
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

BAB IV
IMPLEMENTASI

Setelah melewati proses perancangan, maka dilakukan implementasi sistem. Bab ini akan membahas mengenai implementasi sistem yang meliputi tahap-tahap perancangan program yang telah dibahas pada Bab III. Implementasi kode program dilakukan menggunakan MATLAB.

4.1 Lingkungan Implementasi

Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam implementasi ini ditampilkan pada Tabel 4.1.

Tabel 4.1 Lingkungan Perancangan Perangkat Lunak.

Perangkat	Spesifikasi
Perangkat keras	Prosesor: Intel® Core™ i3 CPU-3240 @ 3.40GHz (4CPUs), ~3.4GHz Memori: 4096 MB
Perangkat lunak	Sistem Operasi: Microsoft Windows 8 Pro 64-bit Perangkat Pengembang: Matlab 2015a Perangkat Pembantu: Microsoft Excel 2013

4.2 Implementasi

Sub bab implementasi ini menjelaskan tentang implementasi proses dari perangkat lunak yang sudah dijelaskan pada bab perancangan perangkat lunak. Implementasi sistem perangkat lunak meliputi implementasi tahap *preprocessing* yang terdiri dari proses normalisasi dan *filtering*, ekstraksi fitur menggunakan metode *autoregressive (AR) model*, klasifikasi

menggunakan *support vector machine*, penentuan kelas final menggunakan metode *majority vote*, dan perhitungan performa perangkat lunak yang terdiri dari perhitungan akurasi, presisi, dan sensitivitas.

4.2.1 Implementasi Tahap *Preprocessing*

Sebelum masuk ke dalam tahap *preprocessing,* label kelas yang semula terdiri dari 6 kelas (label 0 untuk kelas bangun, label 1 untuk kelas NREM1, label 2 untuk kelas NREM2, label 3 untuk kelas NREM3, label 4 untuk kelas NREM4, label 5 untuk kelas REM) akan diubah hanya menjadi 2 kelas yaitu label 0 untuk kelas bangun, label 1 untuk kelas tidur (meliputi NREM1, NREM2, NREM3, NREM4, dan REM). Kode program 4.1 baris 9 sampai 14 merupakan kode program untuk mengubah kelas yang terdiri dari 6 label menjadi 2 label.

Proses pertama pada tahap *preprocessing* adalah normalisasi data dengan *range* [-1 1]. Kode program 4.1 baris 15 sampai 19 merupakan implementasi proses normalisasi. Data hasil normalisasi kemudian di-*filter* menggunakan *butterworth bandpass filtering* untuk memisahkan ke dalam tiga sub band yaitu *thetaband*, *alphaband*, dan *betaband*.

Implementasi desain *butterworth bandpass filtering* dibuat dengan memanggil fungsi *butter* dengan parameter *order = 8, cutoff frequency1 = 4, cutoff frequency2 = 7* untuk *thetaband*, *order = 10, cutoff frequency1 = 8, cutoff frequency2 = 12* untuk *alphaband, order = 11, cutoff frequency1 = 13, cutoff frequency2 = 49* untuk *betaband,* dengan *frequency rate* untuk masing-masing subband adalah 100.

Karena parameter *cutoff frequency1* dan *cutoff frequency2* pada fungsi *butter* harus dalam π *rad/sample*, maka dua parameter tersbut harus dibagi dengan *frequency rate/2*. Kode program 4.1 baris 20 sampai 28 merupakan implementasi proses *butterwaorth bandpass filtering*.

```
1.  filename = dir(strcat(foldername,'*.txt'));
2.  %untuk setiap file dalam 1 subjek
3.  for indexfile = 1:length(filename)
4.    file = strcat(foldername,
      filename(indexfile).name);
5.    delimiterIn = '\t';
6.    data = importdata(file, delimiterIn);
7.    %signal adalah variabel untuk menyimpan
      data EEG channel Fpz-Cz
8.    signal(1,:) = data(:,1);
9.    %simpan label kelas baru ke dalam variabel
      hypnogram
10.   if(data(1,3) == 0)
11.     hypnogram(indexfile,1) = 0;
12.   else
13.     hypnogram(indexfile,1) = 1;
14.   end
15.   %Normalization [-1, 1]
16.   range = max(signal) - min(signal);
17.   for i = 1:length(signal)
18.     signalnorm(1,i) = ((((signal(1,i) -
        min(signal))) / range)*2) - 1;
19.   end
20.   %Desain Butterworth Bandpass Filtering
21.   [a,b] = butter(8, [4 7]/50, 'bandpass');
22.   [c,d] = butter(10, [8 12]/50, 'bandpass');
23.   [e,f] = butter(11, [13 49]/50,
      'bandpass');
24.   %mendapatkan nilai masing-masing subband
25.   thetaband(indexfile,:) =
      filter(a,b,signalnorm(1,:));
26.   alphaband(indexfile,:) =
      filter(c,d,signalnorm(1,:));
27.   betaband(indexfile,:) =
      filter(e,f,signalnorm(1,:));
28. end
```

Kode sumber 4.1 Implementasi Tahap *Preprocessing*.

Variabel *a* dan *b* secara berurutan merupakan vektor koefisien *denominator* (penyebut) dan *numerator* (pembilang) *transfer function* untuk *theta*. Variabel *c* dan *d* secara berurutan merupakan vektor koefisien *denominator* (penyebut) dan *numerator* (pembilang) *transfer function* untuk *alpha*. Variabel *e* dan *f* secara berurutan merupakan vektor koefisien *denominator* (penyebut) dan *numerator* (pembilang) *transfer function* untuk *beta*. *Transfer function* tersebut yang digunakan untuk menyaring data sinyal sesuai dengan batas frekuensi yang diinginkan.

4.2.2 Implementasi Tahap Ekstraksi Fitur

Setelah data EEG dipisahkan menjadi *thetaband, alphaband,* dan *betaband*, proses selanjutnya adalah ekstraksi fitur untuk masing-masing subband. Metode ekstraksi fitur yang digunakan adalah *autoregressive (AR) model* dimana koefisien AR model merupakan fitur yang digunakan pada tahap klasifikasi. Metode yang digunakan untuk mengestimasi koefisien AR model ada 2 yaitu *Yule-Walker* dan *Burg's method*. Kode program 4.2 dan 4.3 merupakan implementasi metode *Yule-Walker*. Kode program 4.4 dan 4.5 merupakan implementasi metode *Burg*.

```
1.  %variabel alpha merupakan hasil estimasi
    koefisien AR model
2.  function alpha = ary(data, order)
3.  N = length(data);
4.  %variable autocorrelation Rxx(k)
5.  C = zeros(order+1, 1);
6.  hasil = zeros(order+1,1);
7.  %menghitung autocorrelation
8.  for k=0:order
9.    hasil(k+1,1) = 0;
10.   for t=k+1:N
11.     hasil(k+1,1) = hasil(k+1,1) +
        data(t)*data(t-k);
```

Kode sumber 4.2 Implementasi Metode Yule-Walker Bagian 1.

```
12.   end
13.   C(k+1,1) = hasil(k+1,1)/N;
14.  end
15.  %variable untuk menyimpan Rxx(0),…,
16.  %Rxx(order)
17.  rx = C(1:end,1);
18.  rx = double(rx);
19.  %levinson-durbin recursion
20.  %inisialisasi
21.  alpha = 1;
22.  error = rx(1,1);
23.  for j=1:order
24.   if(j==1)
25.    gamma = rx(2,1);
26.   else
27.    sum = 0;
28.    for i=2:j
19.     sum = sum + alpha (i,1)*rx(j-i+2,1);
30.    end
31.     gamma = rx(j+1,1) + sum;
32.   end
33.   R = -1*gamma/error;
34.   alpha = [alpha;0] +
      R*[0;conj(flipud(alpha))];
35.   error = error*(1-(abs(R)*abs(R)));
36.  end
```

Kode sumber 4.3 Implementasi Metode *Yule-Walker* Bagian 2.

```
1.   %variabel alpha merupakan hasil estimasi
     koefisien AR model
2.   function alpha = arb(signal, order)
3.   %inisialisasi
4.   alpha = zeros(order+1,1);
5.   ef = signal;
6.   eb = signal;
7.   a = ones(order,order);
8.   k = zeros(order,1);
```

Kode sumber 4. 4 Implementasi Metode Burg Bagian 1.

```
9.  for p=1:order
10.   %Hitung nilai koefisien reflection kp
11.   temp1 = ef(2:end);
12.   temp2 = eb(1:end-1);
13.   numerator = -2 *
      (dot(temp1,conj(temp2)));
14.   denumerator = dot(temp1,temp1) +
      dot(temp2,temp2);
15.   k(p,1) = numerator / denumerator;
16.   %perbaharui nilai ef dan eb
17    ef = temp1 + k(p,1).*temp2;
18.   eb = temp2 + conj(k(p,1)).*temp1;
19.   for i=1:p
20.    if(i==p)
21.     a(p,i) = k(p,1);
22.    else
23.     a(p,i) = a(p-1,i) + k(p,1).*conj(a(p-
        1,p-i));
24.    end
25.   end
26. end
27. alpha (1,1) = 1;
28. alpha (2:end,1) = a(p,:);
29. %transpose variable alpha
30  alpha = alpha.';
```

Kode sumber 4.5 Implementasi Metode *Burg* Bagian 2.

Pada fungsi *main*, untuk mendapatkan atribut *theta, alpha,* dan *beta*, dilakukan pemanggilan fungsi *ary* atau *arb*. Dengan parameter masukan yaitu data pada masing-masing *subband* dan *order* AR model. Keluaran dari fungsi *ary* atau *arb* merupakan koefisien estimasi AR model dengan jumlah *order+1* dimana koefisien awal bernilai 1. Atribut *theta, alpha,* dan *beta* disimpan pada variabel *theta_attributes, alpa_attributes,* dan *beta_attributes*. Kode program 4.6 dan 4.7 merupakan pemanggilan fungsi *ary* dan *arb* secara berurutan untuk mendapatkan atribut *theta, alpha,* dan *beta* pada fungsi *main*.

```
1.  % tahap ekstraksi fitur metode Yule-Walker
2.  theta_attributes(indexfile,:) =
    ary(thetaband(indexfile,:), order_ar);
3.  alpha_attributes(indexfile,:) =
    ary(alphaband(indexfile,:), order_ar);
4.  beta_attributes(indexfile,:) =
    ary(betaband(indexfile,:), order_ar);
```

Kode sumber 4.6 Implementasi Ekstraksi Fitur Metode Yule-Walker pada Fungsi Main.

```
1.  % tahap ekstraksi fitur metode Burg
2.  theta_attributes(indexfile,:) =
    arb(thetaband(indexfile,:), order_ar);
3.  alpha_attributes(indexfile,:) =
    arb(alphaband(indexfile,:), order_ar);
4.  beta_attributes(indexfile,:) =
    arb(betaband(indexfile,:), order_ar);
```

Kode sumber 4.7 Implementasi Ekstraksi Fitur Metode Burg pada Fungsi *Main*.

4.2.3 Implementasi Tahap Klasifikasi

Tahap setelah ekstraksi fitur adalah klasifikasi menggunakan metode *support vector machine*. Atribut *theta, alpha,* dan *beta* dengan indeks 2 sampai *order+1* merupakan masukan pada tahap klasifikasi. Keluaran tahap klasifikasi yaitu kelas (0 berarti bangun, 1 berarti tidur) untuk masing-masing *subband*. Sebelum dilakukan klasifikasi, data masukan dibagi ke dalam 2 kategori yaitu data untuk *training* dan data untuk *testing.* Data *training* dan data *testing* dipisahkan menggunkaan metode *K-fold cross validation* dengan memanggil fungsi *cvpartition* yang sudah tersedia pada MATLAB. Kode program 4.8 adalah kode program untuk memisahkan data menjadi data *training* dan data *testing*. Atribut *theta, alpha*, *beta* dan label kelas data *training* secara berurutan disimpan pada variabel *theta_attributes_training, alpha_attributes_training, beta_attributes_training*, dan *hypno_*

training. Sedangkan atribut *theta, alpha, beta* dan label kelas data *testing* secara berurutan disimpan pada variabel *theta_attributes_testing,alpha_attributes_testing, beta_attributes_ testing*, dan *hypno_ testing*.

```
1.  %inisialisasi 10-fold cross validation
2.  K = 10;
3.  partition =
    cvpartition(length(filename),'kfold',K);
4.  for i = 1:K
5.    %membagi data menjadi data latih dan data uji
6.    loop = training(partition, i);
7.    index_training = 0;
8.    index_testing = 0;
9.    for j = 1:length(filename)
10.     if(loop(j) == 1)
11.       index_training = index_training + 1;
12.       theta_attributes_training(index_training,:)
          = theta_attributes(j,2:end);
13.       alpha_attributes_training(index_training,:)
          = alpha_attributes(j,2:end);
14.       beta_attributes_training(index_training,:)
          = beta_attributes(j,2:end);
15.       hypno_training(index_training,1) =
          hypnogram (j,1);
16.     else
17.       index_testing = index_testing + 1;
18.       theta_attributes_testing(index_testing,:) =
          theta_attributes(j,2:end);
19.       alpha_attributes_testing(index_testing,:) =
          alpha_attributes(j,2:end);
20.       beta_attributes_testing(index_testing,:) =
          beta_attributes(j,2:end);
21.       hypno_testing(index_testing,1) =
          hypnogram(j,1);
22.   end
23. end
```

Kode sumber 4.8 Implementasi 10-*Fold Cross Validation.*

Implementasi *support vector machine* dilakukan dengan memanggil fungsi *svmtrain* dan *svmclassify* yang sudah tersedia pada MATLAB. Data *training* yang telah diperoleh pada tahap sebelumnya dan kelas atau label data *training* menjadi masukan pada fungsi *svmtrain.* Keluaran fungsi *svmtrain* adalah struktur SVM yang berfungsi sebagai *classifier*. Struktur SVM dan data *testing* merukapan masukan pada fungsi *svmclassify.* Kode program 4.9 adalah implementasi klasifikasi menggunakan *support vector machine.* Prediksi kelas untuk *theta, alpha,* dan *beta* secara berurutan disimpan pada variabel *classtheta, classalpha, classbeta*.

```
1.  %Membuat struktur SVM menggunakan svmtrain
2.  SVMStruct_theta =
    svmtrain(theta_attributes_training(:,:),
    hypno_training);
3.  SVMStruct_alpha =
    svmtrain(alpha_attributes_training(:,:),
    hypno_training);
4.  SVMStruct_beta =
    svmtrain(beta_attributes_training(:,:),
    hypno_training);
5.  %Klasifikasi data testing
6.  classtheta = svmclassify(SVMStruct_theta,
    theta_attributes_testing(:,:));
7.  classalpha = svmclassify(SVMStruct_alpha,
    alpha_attributes_testing(:,:));
8.  classbeta = svmclassify(SVMStruct_beta,
    beta_attributes_testing(:,:));
```

Kode sumber 4.9 Implementasi *Support Vector Machine* pada Tahap Klasifikasi.

4.2.4 Implementasi Tahap *Majority Vote*

Setelah masing-masing *subband* diketahui prediksi kelasnya, maka akan dilakukan *majority vote* untuk mengetahui prediksi kelas final. Kode program 4.10 merupakan implmentasi

majority vote. Variabel *class* merupakan variabel yang digunakan untuk menyimpan prediksi final kelas.

```
1.  %menentukan final kelas menggunakan
2.  %majority vote
3.  class = zeros(partition.TestSize(i),1);
4.  sum = 0;
5.  for index = 1:partition.TestSize(i)
6.    sum = classtheta(index,1) +
      classalpha(index,1) + classbeta(index,1);
7.    if(sum < 2)
8.      class(index,1) = 0;
9.    else
10.     class(index,1) = 1;
11.   end
12. end
```

Kode sumber 4.10 Implementasi *Majority Vote* untuk Menentukan Kelas Final.

4.2.5 Implementasi Perhitungan Performa Perangkat Lunak

Perhitungan performa perangkat lunak meliputi perhitungan akurasi, presisi, dan sensitivitas Kode program 4.11 dan 4.12 merupakan implementasi perhitungan performa perangkat lunak.

```
1.  %menghitung performa untuk setiap iterasi
    dari cross validation
2.  confusion_matrix = zeros(2,2);
3.  for z = 1:partition.TestSize(i)
4.    %menghitung true positive
5.    if (hypno_testing(z)==1 && class(z)==1)
6.      confusion_matrix(1,1) =
        confusion_matrix(1,1) + 1;
```

Kode sumber 4.11 Implementasi Perhitungan Performa Perangkat Lunak Bagian 1

```
7.    %menghitung false negative
8.    elseif (hypno_testing(z)==1 &&
      class(z)==0)
9.      confusion_matrix(1,2) =
        confusion_matrix(1,2) + 1;
10.   %menghitung false positive
11.   elseif (hypno_testing(z)==0 &&
      class(z)==1)
12.     confusion_matrix(2,1) =
        confusion_matrix(2,1) + 1;
13.   %menghitung true negative
14.   elseif (hypno_testing(z)==0 &&
      class(z)==0)
15.     confusion_matrix(2,2) =
        confusion_matrix(2,2) + 1;
16.   end
17.   accuracy(i,1) =
      (confusion(1,1)+confusion(2,2))/
      (confusion(1,1)+confusion(1,2)
      +confusion(2,1)+confusion(2,2));
18.   precision(i,1) = confusion(1,1)/
      (confusion(1,1)+confusion(2,1));
19.   sensitivity(i,1) = confusion(1,1)/
      (confusion(1,1)+confusion(1,2));
```

Kode sumber 4.12 Implementasi Perhitungan Performa Perangkat Lunak Bagian 2

(Halaman ini sengaja dikosongkan)

BAB V
UJI COBA DAN EVALUASI

Pada bab ini dijelaskan mengenai rangkaian uji coba perangkat lunak. Pada bab ini juga akan dibahas mengenai perhitungan akurasi dari perangkat lunak yang dibuat. Tujuan dari uji coba ini adalah untuk mengetahui apakah model yang telah dihasilkan sistem mampu mengklasifikasikan keadaan bangun dan tidur dari data sinyal EEG dengan benar, membandingkan metode *Yule-Walker* dan *Burg* untuk mengestimasi koefisien *autoregressive model* pada tahap ekstraksi fitur, mengetahui berapa nilai *order autoregressive model* yang optimal pada tahap ekstraksi fitur, serta mengetahui fungsi kernel pada *support vector machine* yang optimal untuk melakukan klasifikasi. Selain itu juga pada bab ini akan dipaparkan pembahasan yang meliputi lingkungan uji coba, data uji coba, skenario uji coba, hasil uji coba, dan evaluasi.

5.1 Lingkungan Uji Coba

Dibawah ini akan dibahas mengenai lingkungan uji coba perangkat lunak ini, meliputi:

- Perangkat keras yang digunakan untuk uji coba perangkat lunak ini menggunakan komputer dengan prosesor Intel® Core™ i3 CPU M 430 @ 2.27GHz 2.27GHz dengan memori sebesar 2.00 GB.
- Perangkat lunak lingkungan uji coba adalah sistem operasi Windows 7 64-bit dengan perangkat pengembang yang digunakan adalah Matlab 2015a, dan Microsoft Excel 2010.

5.2 Data Uji Coba

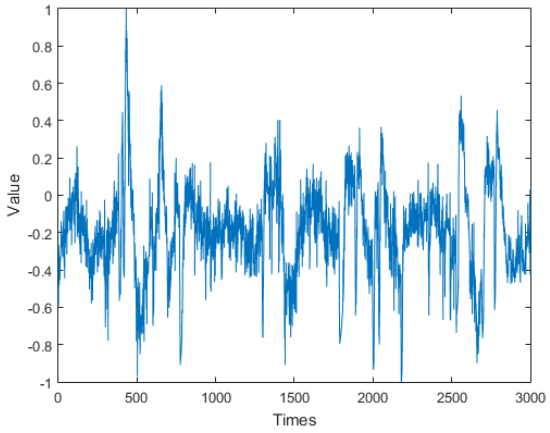
Data uji coba yang digunakan pada perangkat lunak ini adalah data sinyal EEG *sleep* yang tersedia *online* (www.physionet.org) yang sudah dijelaskan pada subbab 3.1.

Dataset terdiri dari 8 subjek dimana setiap subjek datanya telah dibagi menjadi berdurasi 30 detik berdasarkan kelas pada *file hypnogram*.

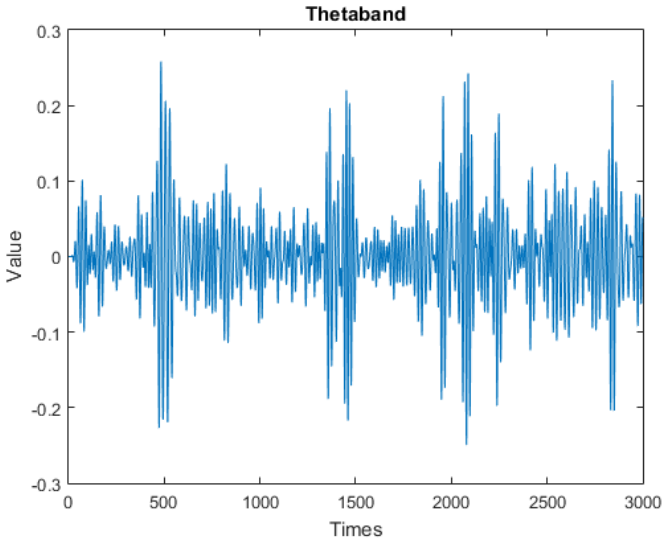
Uji coba dilakukan pada setiap subjek menggunakan skema 10 *fold cross-validation*, dimana setiap dataset semua percobaan dibagi menjadi sepuluh bagian yang berjumlah relatif sama. Sembilan bagian digunakan untuk data latih dan satu bagian sisanya digunakan untuk data uji. Hasil uji coba pada bab ini merupakan hasil rata-rata dari sepuluh iterasi 10 *fold cross-validation*.

5.3 *Preprocessing* Data

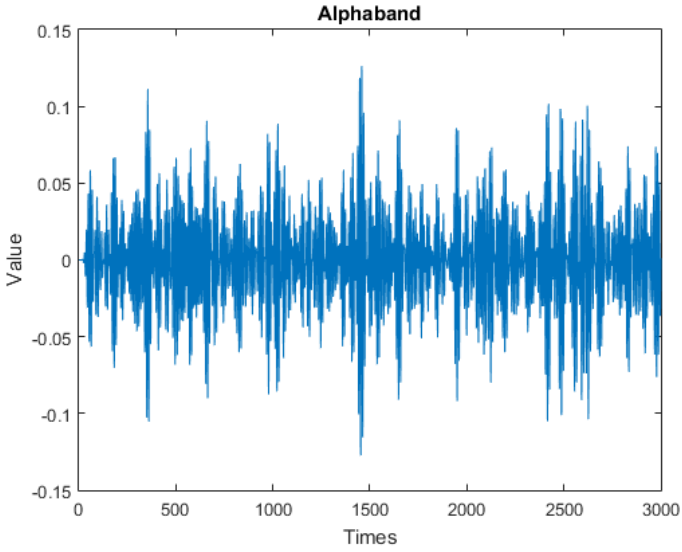
Pada tahap *preprocessing*, data masukan berupa data EEG *channel Fpz-Cz* dinormalisasi dengan *range* [-1 1] kemudian dilakukan *filtering* menjadi tiga *sub band*, yaitu *theta band, alpha band,* dan *beta band.* Gambar 5.1 merupakan contoh hasil dari proses normalisasi data kelas bangun (*awake*) dan secara berurutan data hasil *filtering theta band, alpha band,* dan *beta band* data Gambar 5.1 ditunjukkan pada Gambar 5.2, 5.3, dan 5.4.



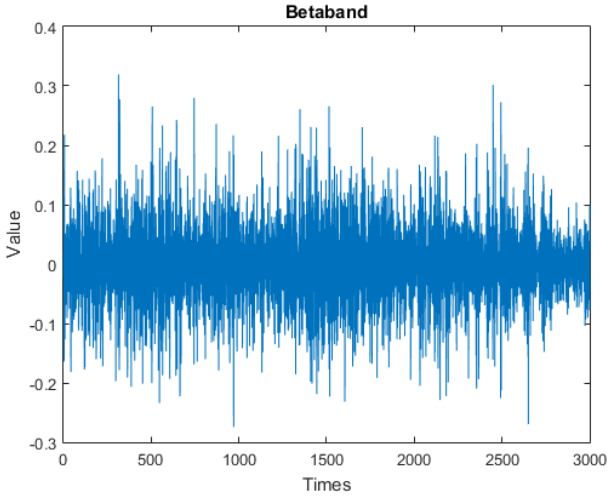
Gambar 5.1 Contoh Data EEG Setelah Dinormalisasi.



Gambar 5.2 Contoh Data *Thetaband.*



Gambar 5.3 Contoh Data *Alphaband.*



Gambar 5.4 Contoh Data *Betaband.*

5.4 Skenario Uji Coba

Melalui skenario ini, perangkat akan diuji apakah sudah berjalan dengan benar dan bagaimana performa masing-masing skenario, serta membandingkan skenario manakah yang memiliki performa lebih baik. Hasil uji coba pada sub bab ini adalah hasil rata-rata dari delapan subjek. Pada setiap skenario uji coba dilakukan perbandingan performa metode *Yule-Walker* dan *Burg*. Hasil setiap subjek pada masing-masing skenario uji coba terdapat pada lampiran. Terdapat tiga skenario uji coba pada perangkat lunak ini, antara lain:

1. Perhitungan performa dengan mengubah nilai *order (p) autoregressive model* pada proses ekstraksi fitur. Nilai *order* yang diujikan antara lain: 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100.
2. Perhitungan performa dengan variasi fungsi *kernel*. Fungsi *kernel* yang diujikan yaitu *linear* dan RBF (*Gaussian Radial Basis Function*) dengan nilai *gamma* antara lain: 0.5, 1, 1.5, dan 2.

3. Perhitungan performa dengan mengubah nilai parameter C pada SVM. Nilai parameter C yang diujikan antara lain: 1, 2, 3, 4, dan 5.

5.4.1 Skenario Uji Coba 1

Skenario uji coba pertama adalah perhitungan performa yang meliputi akurasi, presisi, sensitivitas dengan variasi nilai *order (p) autoregressive model*. Nilai *p* yang diuji coba antara lain: 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100. Nilai order AR merepresentasikan banyaknya fitur yang digunakan pada tahap klasifikasi. Pada skenario uji coba ini, jenis *kernel function* yang digunakan pada SVM adalah RBF (*Radial Basis Function*) dengan parameter *gamma=1*. Hasil performa dari masing-masing nilai *p* dapat dilihat pada Tabel 5.1, 5.2, dan 5.3.

Tabel 5.1 Hasil Akurasi Uji Coba Penggantian Nilai *Order* AR Model.

Order AR	Metode *Yule-Walker* (%)	Metode *Burg* (%)
10	73.91	89.60
20	77.80	92.58
30	78.12	93.27
40	78.82	93.90
50	**81.21**	94.13
60	79.77	94.58
70	80.91	**94.84**
80	78.85	94.20
90	80.04	93.47
100	79.03	91.17
rata-rata	**78.85**	**93.17**

Berdasarkan hasil uji coba perhitungan akurasi yang ditunjukkan pada Tabel 5.1, metode *Yule-Walker* mencapai nilai akurasi tertinggi 81.21% ketika *order* = 50 sedangkan metode *Burg* mencapai nilai akurasi tertinggi 94.84% ketika *order* = 70. Rata – rata akurasi metode *Yule-Walker* adalah 78.85%, sedangkan rata – rata akurasi metode *Burg* adalah 93.17%.

Tabel 5.2 Hasil Presisi Uji Coba Penggantian Nilai *Order* AR Model.

Order AR	Metode *Yule-Walker* (%)	Metode *Burg* (%)
10	75.32	93.17
20	77.79	**97.08**
30	78.19	97.06
40	77.22	96.83
50	77.19	96.75
60	78.00	95.58
70	**78.33**	95.38
80	77.30	94.60
90	77.32	94.40
100	77.45	95.21
rata-rata	**77.41**	**95.61**

Berdasarkan hasil uji coba perhitungan presisi yang ditunjukkan pada Tabel 5.2, metode *Yule-Walker* mencapai nilai presisi tertinggi 78.33% ketika *order* = 70 sedangkan metode *Burg* mencapai nilai presisi tertinggi 97.08% ketika *order* = 70. Rata – rata presisi metode *Yule-Walker* adalah 77.41%, sedangkan rata – rata presisi metode *Burg* adalah 95.61%.

Tabel 5.3 Hasil Sensitivitas Uji Coba Penggantian Nilai *Order* AR Model.

Order AR	Metode *Yule-Walker* (%)	Metode *Burg* (%)
10	78.20	87.21
20	78.51	89.96
30	77.38	91.39
40	78.75	92.26
50	**80.47**	92.80
60	77.51	94.46
70	77.91	**95.40**
80	75.51	95.28
90	76.99	93.44
100	75.18	85.67
rata-rata	**77.64**	**91.79**

Berdasarkan hasil uji coba yang ditunjukkan pada perhitungan sensitivitas di atas, metode *Yule-Walker* mencapai nilai sensitivitas tertinggi 80.47% ketika *order* = 50 sedangkan metode *Burg* mencapai nilai sensitivitas tertinggi 95.40% ketika *order* = 70. Rata – rata sensitivitas metode *Yule-Walker* adalah 77.64%, sedangkan rata – rata sensitivitas metode *Burg* adalah 91.79%.

5.4.2 Skenario Uji Coba 2

Skenario uji coba kedua adalah perhitungan performa yang meliputi akurasi, presisi, sensitivitas dengan memvariasikan fungsi *kernel support vector machine*. Fungsi *kernel* yang diuji coba

antara lain: *linear,* RBF (*Radian Basis Function*) dengan nilai *gamma* antara lain: 0.5, 1, 1.5, dan 2.

Nilai *gamma* pada RBF mempengaruhi besar kecilnya *variance*. Semakin besar nilai *gamma* mengakibatkan semakin kecil nilai *variance*, sehingga *support vectors* memiliki pengaruh yang besar dalam menentukan kelas dari data x_i. Sebaliknya, jika nilai *gamma* semakin kecil mengakibatkan semakin besar nilai *variance*, sehingga pengaruh *support vectors* dalam menentukan kelas dari data x_i sangat kecil.

Pada skenario uji coba ini, nilai parameter C pada *support vector machine* yang digunakan adalah 1. *Order autoregressive (AR) model* yang digunakan untuk metode *Yule-Walker* adalah 50 dan 70 untuk metode *Burg*. Penentuan nilai *order* tersebut didasarkan pada hasil uji coba pada skenario 1, dimana nilai akurasi dan sensitivitas tertinggi untuk metode *Yule-Walker* dicapai ketika *order autoregressive (AR) model* = 50, dan metode *Burg* dicapai ketika *order autoregressive (AR) model* = 70. Hasil performa dari masing-masing variasi fungsi *kernel* dapat dilihat pada Tabel 5.4 dan 5.5.

Tabel 5.4 Hasil Uji Coba Variasi Fungsi *Kernel* SVM dengan Metode *Yule-Walker.*

fungsi *kernel*	akurasi (%)	presisi (%)	sensitivitas (%)
linear	78.47	78.46	78.71
RBF (gamma=0.5)	**82.96**	78.76	75.96
RBF (gamma=1)	78.30	76.21	80.54
RBF (gamma=1.5)	79.84	**76.78**	**81.10**
RBF (gamma=2)	78.48	76.15	81.02
rata-rata	**79.61**	**77.27**	**79.47**

Tabel 5.5 Hasil Uji Coba Variasi Fungsi *Kernel* dengan Metode *Burg*.

fungsi *kernel*	akurasi (%)	presisi (%)	sensitivitas (%)
linear	93.51	**97.38**	92.05
RBF (gamma=0.5)	92.64	94.64	89.90
RBF (gamma=1)	**94.93**	95.46	**95.41**
RBF (gamma=1.5)	94.88	96.64	94.09
RBF (gamma=2)	94.35	96.98	93.10
rata-rata	**94.06**	**96.22**	**92.91**

Berdasarkan hasil uji coba yang ditunjukkan pada perhitungan performa di atas, metode *Yule-Walker* mencapai nilai akurasi tertinggi ketika fungsi *kernel* yang digunakan pada SVM adalah RBF dengan nilai *gamma*=0.5, dan mencapai nilai sensitivitas dan presisi tertinggi ketika fungsi *kernel* yang digunakan pada SVM adalah RBF dengan nilai *gamma*=1.5. Sedangkan metode *Burg* mencapai nilai akurasi dan sensitivitas tertinggi ketika fungsi *kernel* yang digunakan pada SVM adalah RBF dengan nilai *gamma*=1, dan mencapai nilai presisi tertinggi ketika fungsi *kernel* yang digunakan pada SVM adalah *linear*. Rata-rata akurasi, presisi, dan sensitivitas metode *Yule-Walker* berturut-turut adalah 79.61%, 77.27%, dan 79.47%. Rata-rata akurasi, presisi, dan sensitivitas metode *Burg* berturut-turut adalah 94.06%, 96.22%, dan 92.91%.

5.4.3 Skenario Uji Coba 3

Skenario uji coba ketiga adalah perhitungan performa yang meliputi akurasi, presisi, sensitivitas dengan variasi nilai parameter C pada *support vector machine*. Nilai parameter C yang diujikan antara lain: 1, 2, 3, 4, dan 5. Nilai parameter C mempengaruhi besar kecilnya *margin hyperplane.* Semakin besar nilai C, maka *margin*

hyperplane akan semakin kecil. Sebaliknya, nilai C yang besar mengakibatkan semakin kecilnya *margin hyperplane*.

Pada skenario uji coba ini, *Order* AR model yang digunakan untuk metode *Yule-Walker* adalah 50, dan 70 untuk metode *Burg*. Fungsi *kernel* SVM yang digunakan adalah RBF dengan *gamma*=0.5 untuk metode *Yule-Walker*, sedangkan untuk metode *Burg* digunakan RBF dengan *gamma*=1. Hasil performa dari skenario uji coba ini dapat dilihat pada Tabel 5.6 dan 5.7.

Tabel 5.6 Hasil Uji Coba Variasi Nilai Parameter C dengan Metode *Yule-Walker*.

Nilai C	akurasi (%)	presisi (%)	sensitivitas (%)
1	82.96	78.76	**75.96**
2	82.84	79.27	74.98
3	83.29	79.88	75.69
4	83.67	80.44	75.61
5	**83.93**	**81.06**	75.54
rata-rata	**83.34**	**79.88**	**75.56**

Tabel 5.7 Hasil Uji Coba Variasi Nilai Parameter C dengan Metode *Burg*.

Nilai C	akurasi (%)	presisi (%)	sensitivitas (%)
1	**94.93**	**95.46**	95.41
2	94.81	95.29	95.37
3	94.83	95.26	**95.57**
4	94.70	95.11	95.49
5	94.77	95.20	95.44
rata-rata	**94.81**	**95.26**	**95.46**

Berdasarkan hasil uji coba yang ditunjukkan pada perhitungan performa di atas, metode *Yule-Walker* dan fungsi *kernel* RBF dengan gamma=0.5 mencapai nilai akurasi dan presisi tertinggi ketika nilai parameter C=5, dan mencapai nilai sensitivitas tertinggi ketika nilai parameter C=1. Sedangkan metode *Burg* dan fungsi *kernel* RBF dengan gamma=1 mencapai nilai akurasi dan presisi tertinggi ketika nilai parameter C=1, dan mencapai nilai sensitivitas tertinggi ketika nilai parameter C=3. Rata-rata akurasi, presisi, dan sensitivitas metode *Yule-Walker* berturut-turut adalah 83.34%, 79.88%, dan 75.56%. Rata-rata akurasi, presisi, dan sensitivitas metode *Burg* berturut-turut adalah 94.81%, 95.26%, dan 5.46%.

5.5 Evaluasi

Dari hasil skenario uji coba yang telah dilakukan, dapat diketahui bahwa performa sistem dalam mengklasifikasikan data EEG untuk mendeteksi keadaan tidur dan bangun akan lebih optimal jika dalam ekstraksi fitur digunakan metode *Burg* daripada metode *Yule-Walker* untuk mengestimasi koefisien AR model. Hal tersebut terbukti pada ketiga skenario uji coba menghasilkan rata-rata akurasi 80.60%, presisi 78.19%, sensitivitas 77.56% untuk metode *Yule-Walker*. Sedangkan untuk metode *Burg* menghasilkan rata-rata akurasi 94.01%, presisi 95.70%, dan sensitivitas 93.39%.

Beberapa parameter juga memberikan pengaruh terhadap performa sistem. Parameter yang digunakan antara lain nilai *order* AR model dan nilai C pada SVM. Uji coba dilakukan dengan membandingkan tingkat akurasi, presisi, dan sensitivitas.

Pada skenario uji coba 1, parameter yang diuji adalah nilai *order* AR model. Hasil percaobaan menunjukkan ketika menggunakan metode *Yule-Walker* akurasi tertinggi didapat ketika *order* AR = 50, presisi tertinggi didapat ketika *order* AR = 70, dan sensitivitas tertinggi didapat ketika *order* AR = 50. Ketika menggunakan metode *Burg* akurasi tertinggi didapat ketika *order*

AR = 70, presisi tertinggi didapat ketika *order* AR = 20, dan sensitivitas tertinggi didapat ketika *order* AR = 70.

Pada uji coba 2, uji coba dilakukan dengan mengubah fungsi *kernel* pada SVM. Hasil percobaan menunjukkan untuk metode *Yule-Walker* akurasi tertinggi didapat ketika kernel *fungsi* yang digunakan adalah RBF dengan *gamma*=0.5, presisi tertinggi didapat ketika kernel *fungsi* yang digunakan adalah RBF dengan *gamma*=0.5, dan sensitivitas tertinggi didapat ketika kernel *fungsi* yang digunakan adalah RBF dengan *gamma*=1.5. Untuk metode *Burg,* akurasi tertinggi didapat ketika kernel *fungsi* yang digunakan adalah RBF dengan *gamma*=1, presisi tertinggi didapat ketika kernel *fungsi* yang digunakan adalah *linear*, dan sensitivitas tertinggi didapat ketika kernel *fungsi* yang digunakan adalah RBF dengan *gamma*=1.

Pada uji coba 3, parameter yang diuji adalah nilai C pada SVM. Variasi nilai C yang diujikan antara lain: 1, 2, 3, 4, dan 5. Hasil percobaan menunjukkan untuk metode *Yule-Walker* akurasi tertinggi didapat ketika nilai C=5, presisi tertinggi didapat ketika nilai C=5, dan sensitivitas tertinggi didapat ketika nilai C=1. Untuk metode *Burg* akurasi tertinggi didapat ketika nilai C=1, presisi tertinggi didapat ketika nilai C=1, dan sensitivitas tertinggi didapat ketika nilai C=3.

Dari hasil ketiga skenario dapat disimpulkan bahwa metode untuk mengestimasi koefisien AR dengan mempertimbangkan *error forward prediction* dan *error backward prediction* (metode *Burg*) memiliki performa lebih baik daripada hanya mempertimbangkan koefisien *autocorrelation* (metode *Yule-Walker*).

LAMPIRAN

1. Hasil uji coba variasi nilai *order* AR untuk metode *Yule-Walker* dan *Burg* untuk setiap subjek.

Tabel 7.1 Hasil Uji Coba Order AR = 10 Metode *Yule-Walker.*

subjek	akurasi (%)	precision (%)	recall (%)
1	73.48	57.83	76.78
2	75.45	63.02	79.14
3	71.92	55.54	79.30
4	69.49	43.55	83.09
5	70.96	95.24	72.26
6	82.48	92.43	87.38
7	69.31	97.83	68.70
8	78.17	97.13	78.92
rata-rata	**73.91**	**75.32**	**78.20**

Tabel 7.2 Hasil Uji Coba Order AR = 20 Metode *Yule-Walker.*

subjek	akurasi (%)	precision (%)	recall (%)
1	77.55	64.29	74.79
2	77.66	67.26	74.28
3	74.33	59.84	69.73
4	74.56	48.90	82.42
5	79.26	95.17	81.70
6	82.88	93.72	86.61
7	77.06	97.39	77.54
8	79.11	95.72	81.04
rata-rata	**77.80**	**77.79**	**78.51**

Tabel 7.3 Hasil Uji Coba Order AR = 30 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	77.58	66.07	68.65
2	78.96	70.65	70.94
3	73.81	58.48	72.10
4	74.63	48.74	78.13
5	77.77	93.58	81.39
6	83.95	94.40	87.29
7	85.20	96.32	87.50
8	73.08	97.30	73.04
rata-rata	**78.12**	**78.19**	**77.38**

Tabel 7.4 Hasil Uji Coba Order AR = 40 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	76.98	64.15	70.17
2	77.34	69.24	67.60
3	72.66	56.83	72.76
4	76.14	50.69	77.07
5	82.87	93.43	87.70
6	78.99	93.20	82.74
7	84.71	94.29	88.97
8	80.88	95.97	83.02
rata-rata	**78.82**	**77.22**	**78.75**

Tabel 7.5 Hasil Uji Coba Order AR = 50 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	76.67	64.45	67.62
2	77.91	68.64	71.81
3	73.95	59.54	66.81
4	74.95	49.06	72.08
5	86.06	93.21	91.64
6	88.52	92.00	95.48
7	87.55	94.23	92.36
8	84.06	96.41	85.99
rata-rata	**81.21**	**77.19**	**80.47**

Tabel 7.6 Hasil Uji Coba Order AR = 60 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	77.09	65.62	65.47
2	77.94	69.07	70.03
3	74.23	60.18	66.05
4	77.04	52.25	71.42
5	85.96	93.04	91.74
6	74.81	93.16	77.54
7	90.88	94.60	95.68
8	80.17	96.08	82.12
rata-rata	**79.77**	**78.00**	**77.51**

Tabel 7.7 Hasil Uji Coba Order AR = 70 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	78.82	69.96	64.15
2	77.59	68.92	69.12
3	72.97	58.21	65.24
4	77.65	53.46	66.81
5	85.64	92.98	91.37
6	83.55	92.69	88.55
7	90.88	94.42	95.94
8	80.17	96.00	82.14
rata-rata	**80.91**	**78.33**	**77.91**

Tabel 7.8 Hasil Uji Coba Order AR = 80 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	76.56	65.63	61.62
2	74.46	64.64	63.47
3	73.43	58.91	65.48
4	78.12	54.04	68.11
5	85.21	92.49	91.53
6	79.97	93.54	83.40
7	90.10	93.57	96.04
8	72.95	95.60	74.43
rata-rata	**78.85**	**77.30**	**75.51**

Tabel 7.9 Hasil Uji Coba Order AR = 90 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	76.41	64.59	65.33
2	75.80	66.92	64.86
3	73.84	60.02	62.14
4	76.00	50.40	68.64
5	86.17	93.33	91.64
6	78.40	93.21	81.85
7	91.86	93.92	97.57
8	81.81	96.16	83.84
rata-rata	**80.04**	**77.32**	**76.99**

Tabel 7.10 Hasil Uji Coba Order AR = 100 Metode *Yule-Walker*.

subjek	akurasi (%)	precision (%)	recall (%)
1	76.52	65.48	63.70
2	74.01	64.57	61.46
3	73.77	60.42	60.31
4	78.70	55.20	68.43
5	88.51	92.98	94.69
6	79.57	93.22	83.23
7	91.76	93.49	98.02
8	69.43	94.20	71.56
rata-rata	**79.03**	**77.45**	**75.18**

Tabel 7.11 Hasil Uji Coba Order AR = 10 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	93.99	86.50	97.35
2	88.52	84.46	83.35
3	90.13	92.74	76.08
4	94.49	87.34	90.65
5	90.85	98.47	91.45
6	83.36	98.85	82.41
7	90.49	99.29	90.45
8	85.00	97.72	85.91
rata-rata	**89.60**	**93.17**	**87.21**

Tabel 7.12 Hasil Uji Coba Order AR = 20 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	96.15	92.36	96.32
2	93.19	92.94	87.71
3	93.42	96.67	82.95
4	97.91	97.88	93.38
5	88.30	99.23	88.04
6	90.86	99.27	90.40
7	92.55	99.42	92.48
8	88.31	98.85	88.39
rata-rata	**92.58**	**97.08**	**89.96**

Tabel 7.13 Hasil Uji Coba Order AR = 30 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	95.93	92.30	95.91
2	94.98	92.65	93.40
3	94.01	97.25	84.30
4	98.06	98.27	93.61
5	89.26	98.87	89.46
6	92.03	99.41	91.71
7	93.04	99.02	93.51
8	88.90	98.73	89.20
rata-rata	**93.27**	**97.06**	**91.39**

Tabel 7.14 Hasil Uji Coba Order AR = 40 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	95.90	92.28	95.71
2	95.47	92.99	94.53
3	93.70	95.74	84.84
4	97.91	97.95	93.49
5	92.02	98.89	92.41
6	92.80	99.18	92.74
7	93.92	99.02	94.44
8	89.49	98.60	89.92
rata-rata	**93.90**	**96.83**	**92.26**

Tabel 7.15 Hasil Uji Coba Order AR = 50 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	95.76	91.88	95.57
2	95.82	93.95	94.56
3	94.22	96.31	85.98
4	98.02	98.15	93.76
5	92.66	98.81	93.24
6	93.58	98.44	94.35
7	93.24	98.68	94.01
8	89.73	97.77	90.95
rata-rata	**94.13**	**96.75**	**92.80**

Tabel 7.16 Hasil Uji Coba Order AR = 60 Metode *Burg*.

Subjek	akurasi (%)	precision (%)	recall (%)
1	96.07	92.37	96.05
2	95.82	93.07	95.61
3	93.94	95.55	85.92
4	97.91	97.78	93.52
5	92.98	96.31	96.09
6	93.58	96.91	95.86
7	94.22	96.84	96.94
8	92.09	95.82	95.68
rata-rata	**94.58**	**95.58**	**94.46**

Tabel 7.17 Hasil Uji Coba Order AR = 70 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	96.71	93.84	96.68
2	95.50	93.10	94.53
3	94.85	96.01	88.26
4	98.02	98.58	93.35
5	93.72	95.43	97.96
6	92.70	94.79	97.21
7	93.92	95.41	98.21
8	93.27	95.88	96.96
rata-rata	**94.84**	**95.38**	**95.40**

Tabel 7.18 Hasil Uji Coba Order AR = 80 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	96.43	94.18	95.25
2	95.82	95.09	93.17
3	94.15	96.79	85.23
4	97.81	98.72	92.12
5	92.98	93.70	99.08
6	90.27	91.11	98.82
7	93.82	93.87	99.90
8	92.32	93.38	98.71
rata-rata	**94.20**	**94.60**	**95.28**

Tabel 7.19 Hasil Uji Coba Order AR = 90 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	96.00	94.89	93.00
2	94.45	94.92	89.51
3	92.30	97.54	78.78
4	96.58	98.90	86.92
5	92.45	92.44	100.00
6	89.79	89.94	99.78
7	93.63	93.60	100.00
8	92.55	92.98	99.49
rata-rata	**93.47**	**94.40**	**93.44**

Tabel 7.20 Hasil Uji Coba Order AR = 100 Metode *Burg*.

subjek	akurasi (%)	precision (%)	recall (%)
1	95.62	97.57	89.15
2	86.37	98.30	63.18
3	84.63	98.24	55.01
4	94.57	99.37	78.00
5	92.34	92.34	100.00
6	89.68	89.68	100.00
7	93.24	93.24	100.00
8	92.92	92.92	100.00
rata-rata	**91.17**	**95.21**	**85.67**

2. Hasil uji coba variasi fungsi *kernel* SVM untuk metode *Yule-Walker* dengan *order* AR=50 dan metode *Burg* dengan *order* AR=70 untuk setiap subjek.

Tabel 7.21 Hasil Akurasi (%) Uji Coba Variasi Fungsi *Kernel* SVM untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ=0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	77.55	75.99	76.59	76.52	76.70
2	79.87	75.59	76.40	76.92	75.76
3	76.44	75.07	71.60	73.77	71.60
4	68.48	78.84	72.72	74.16	72.58
5	79.89	90.64	81.60	82.77	83.83
6	88.53	87.26	89.69	90.75	89.69
7	84.02	92.35	78.73	83.04	77.55
8	72.98	87.96	79.10	80.76	80.17
rata-rata	**78.47**	**82.96**	**78.30**	**79.84**	**78.48**

Tabel 7.22 Hasil Presisi (%) Uji Coba Variasi Fungsi *Kernel* SVM untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	65.36	67.13	63.01	63.00	62.70
2	72.07	69.34	66.42	66.24	65.32
3	63.87	64.52	55.43	58.41	55.61
4	43.43	56.51	46.59	48.00	46.64
5	95.94	92.40	93.74	93.71	94.00
6	91.89	91.99	92.00	92.37	92.15
7	97.19	93.43	95.43	95.51	95.86
8	97.94	94.79	97.09	96.99	96.93
rata-rata	**78.46**	**78.76**	**76.21**	**76.78**	**76.15**

Tabel 7.23 Hasil Sensitivitas (%) Uji Coba Variasi fungsi *Kernel* SVM untuk Metode *Yule-Walker.*

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	68.72	55.95	73.24	72.18	75.06
2	73.06	57.72	70.02	73.65	70.33
3	66.25	55.86	74.47	72.66	75.68
4	86.68	55.52	82.82	77.61	81.71
5	81.69	97.94	85.85	87.25	88.13
6	95.69	93.89	96.96	97.84	96.73
7	85.19	98.74	81.09	85.78	79.43
8	72.42	92.10	79.88	81.85	81.13
rata-rata	**78.71**	**75.96**	**80.54**	**81.10**	**81.02**

Tabel 7.24 Hasil Akurasi (%) Uji Coba Variasi Fungsi *Kernel* SVM untuk Metode *Burg.*

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	96.11	96.50	96.64	96.50	96.36
2	96.31	93.04	95.79	95.61	95.82
3	95.80	91.49	94.78	94.96	94.89
4	97.95	91.33	98.02	98.13	98.02
5	90.74	92.34	93.62	94.26	92.87
6	92.12	89.69	92.99	93.78	93.09
7	92.25	93.24	94.12	93.73	94.02
8	86.77	93.51	93.51	92.10	89.74
rata-rata	**93.51**	**92.64**	**94.93**	**94.88**	**94.35**

Tabel 7.25 Hasil Presisi (%) Uji Coba Variasi Fungsi *Kernel* SVM untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	92.54	94.89	93.68	93.53	93.11
2	94.65	95.68	93.89	93.03	93.18
3	95.89	97.86	96.39	96.11	96.43
4	97.62	99.77	98.51	98.46	98.11
5	99.77	92.43	94.99	98.11	98.77
6	99.39	89.69	94.92	97.97	98.62
7	99.67	93.32	95.70	98.16	99.11
8	99.54	93.47	95.63	97.75	98.47
rata-rata	**97.38**	**94.64**	**95.46**	**96.64**	**96.98**

Tabel 7.26 Hasil Sensitivitas (%) Uji Coba Variasi Fungsi *Kernel* SVM untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	95.95	94.48	96.41	96.19	96.17
2	95.30	84.52	94.53	95.02	95.32
3	91.25	75.94	87.64	88.34	87.96
4	93.76	64.46	93.42	93.80	93.57
5	90.25	99.88	98.27	95.63	93.42
6	91.74	100.00	97.41	95.01	93.60
7	92.04	99.89	98.13	95.07	94.42
8	86.13	100.00	97.48	93.64	90.35
rata-rata	**92.05**	**89.90**	**95.41**	**94.09**	**93.10**

3. Hasil uji coba variasi nilai parameter C pada SVM untuk metode *Yule-Walker* dengan *order* AR=50 dan metode *Burg* dengan *order* AR=70 untuk setiap subjek.

Tabel 7.27 Hasil Akurasi (%) Uji Coba dengan Nilai C = 1 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	77.55	75.99	76.59	76.52	76.70
2	79.87	75.59	76.40	76.92	75.76
3	76.44	75.07	71.60	73.77	71.60
4	68.48	78.84	72.72	74.16	72.58
5	79.89	90.64	81.60	82.77	83.83
6	88.53	87.26	89.69	90.75	89.69
7	84.02	92.35	78.73	83.04	77.55
8	72.98	87.96	79.10	80.76	80.17
rata-rata	**78.47**	**82.96**	**78.30**	**79.84**	**78.48**

Tabel 7.28 Hasil Presisi (%) Uji Coba dengan Nilai C = 1 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	65.36	67.13	63.01	63.00	62.70
2	72.07	69.34	66.42	66.24	65.32
3	63.87	64.52	55.43	58.41	55.61
4	43.43	56.51	46.59	48.00	46.64
5	95.94	92.40	93.74	93.71	94.00
6	91.89	91.99	92.00	92.37	92.15
7	97.19	93.43	95.43	95.51	95.86
8	97.94	94.79	97.09	96.99	96.93
rata-rata	**78.46**	**78.76**	**76.21**	**76.78**	**76.15**

Tabel 7.29 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 1 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	68.72	55.95	73.24	72.18	75.06
2	73.06	57.72	70.02	73.65	70.33
3	66.25	55.86	74.47	72.66	75.68
4	86.68	55.52	82.82	77.61	81.71
5	81.69	97.94	85.85	87.25	88.13
6	95.69	93.89	96.96	97.84	96.73
7	85.19	98.74	81.09	85.78	79.43
8	72.42	92.10	79.88	81.85	81.13
rata-rata	**78.71**	**75.96**	**80.54**	**81.10**	**81.02**

Tabel 7.30 Hasil Akurasi (%) Uji Coba dengan Nilai C = 1 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	96.11	96.50	96.64	96.50	96.36
2	96.31	93.04	95.79	95.61	95.82
3	95.80	91.49	94.78	94.96	94.89
4	97.95	91.33	98.02	98.13	98.02
5	90.74	92.34	93.62	94.26	92.87
6	92.12	89.69	92.99	93.78	93.09
7	92.25	93.24	94.12	93.73	94.02
8	86.77	93.51	93.51	92.10	89.74
rata-rata	**93.51**	**92.64**	**94.93**	**94.88**	**94.35**

Tabel 7.31 Hasil presisi (%) Uji Coba dengan Nilai C = 1 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	92.54	94.89	93.68	93.53	93.11
2	94.65	95.68	93.89	93.03	93.18
3	95.89	97.86	96.39	96.11	96.43
4	97.62	99.77	98.51	98.46	98.11
5	99.77	92.43	94.99	98.11	98.77
6	99.39	89.69	94.92	97.97	98.62
7	99.67	93.32	95.70	98.16	99.11
8	99.54	93.47	95.63	97.75	98.47
rata-rata	**97.38**	**94.64**	**95.46**	**96.64**	**96.98**

Tabel 7.32 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 1 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	95.95	94.48	96.41	96.19	96.17
2	95.30	84.52	94.53	95.02	95.32
3	91.25	75.94	87.64	88.34	87.96
4	93.76	64.46	93.42	93.80	93.57
5	90.25	99.88	98.27	95.63	93.42
6	91.74	100.00	97.41	95.01	93.60
7	92.04	99.89	98.13	95.07	94.42
8	86.13	100.00	97.48	93.64	90.35
rata-rata	**92.05**	**89.90**	**95.41**	**94.09**	**93.10**

Tabel 7.33 Hasil Akurasi (%) Uji Coba dengan Nilai C = 2 untuk Metode *Yule-Walker.*

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	78.47	76.13	76.91	76.59	76.45
2	81.07	76.32	79.14	78.68	77.91
3	76.85	74.72	75.10	76.12	72.79
4	73.05	79.52	78.01	75.89	74.85
5	79.47	91.06	86.70	83.72	83.94
6	83.47	83.56	88.13	88.43	90.08
7	82.75	92.25	88.63	85.29	79.80
8	73.80	89.14	83.95	82.88	81.70
rata-rata	**78.62**	**82.84**	**82.07**	**80.95**	**79.69**

Tabel 7.34 Hasil Presisi (%) Uji Coba dengan Nilai C = 2 untuk Metode *Yule-Walker.*

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	67.04	67.15	64.90	63.42	62.52
2	73.48	70.54	69.66	68.71	67.33
3	64.64	64.86	61.36	61.69	56.94
4	47.11	58.93	53.74	50.19	49.11
5	96.22	92.43	92.94	93.55	93.90
6	92.32	92.17	92.97	91.70	92.28
7	97.12	93.25	93.99	95.08	95.40
8	96.84	94.83	95.81	95.91	97.41
rata-rata	**79.34**	**79.27**	**78.17**	**77.53**	**76.86**

Tabel 7.35 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 2 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	70.29	55.20	67.44	71.14	74.08
2	74.43	58.99	73.84	74.93	74.77
3	67.30	52.70	66.83	74.79	76.56
4	85.69	53.16	71.51	75.70	79.57
5	81.00	98.40	92.62	88.47	88.33
6	88.23	89.23	93.78	95.71	97.06
7	84.03	98.84	93.78	88.90	82.36
8	73.96	93.36	86.61	85.29	82.38
rata-rata	**78.12**	**74.98**	**80.80**	**81.87**	**81.89**

Tabel 7.36 Hasil Akurasi (%) Uji Coba dengan Nilai C = 2 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	96.04	96.15	96.71	96.43	96.29
2	96.35	92.80	95.71	95.68	95.89
3	95.76	89.81	94.75	95.17	95.27
4	97.95	91.18	97.91	98.09	98.02
5	90.53	92.34	93.62	94.47	94.04
6	92.22	89.69	92.41	94.37	93.29
7	92.25	93.33	94.02	93.73	94.61
8	87.36	93.28	93.39	92.45	90.67
rata-rata	**93.56**	**92.32**	**94.81**	**95.05**	**94.76**

Tabel 7.37 Hasil Presisi (%) Uji Coba dengan Nilai C = 2 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	92.53	95.24	93.98	93.33	93.11
2	94.65	96.67	94.14	93.45	93.64
3	95.57	97.67	96.30	96.37	96.53
4	97.62	99.75	98.43	98.46	97.94
5	99.64	92.34	95.03	97.55	98.54
6	99.28	89.69	93.72	97.36	98.64
7	99.67	93.42	95.22	97.13	99.13
8	99.70	93.25	95.52	96.89	97.58
rata-rata	**97.33**	**94.75**	**95.29**	**96.32**	**96.89**

Tabel 7.38 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 2 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	95.73	93.06	96.23	96.16	95.99
2	95.38	82.81	93.86	94.73	95.23
3	91.46	71.16	87.65	88.82	88.99
4	93.76	63.93	92.78	93.63	94.05
5	90.13	100.00	98.27	96.44	94.93
6	91.96	100.00	98.17	96.34	93.86
7	92.02	99.90	98.53	96.10	95.08
8	86.63	100.00	97.45	94.93	92.28
rata-rata	**92.13**	**88.86**	**95.37**	**94.64**	**93.80**

Tabel 7.39 Hasil Akurasi (%) Uji Coba dengan Nilai C = 3 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	78.04	76.84	77.26	76.80	77.58
2	82.44	77.63	79.91	81.00	78.82
3	77.66	74.23	76.40	75.45	74.96
4	75.42	80.46	78.84	78.95	76.61
5	77.77	91.17	88.19	84.68	82.55
6	88.23	84.73	87.44	89.60	90.36
7	82.06	92.45	89.22	86.18	83.14
8	74.25	88.80	84.31	81.94	81.70
rata-rata	**79.48**	**83.29**	**82.70**	**81.82**	**80.72**

Tabel 7.40 Hasil Presisi (%) Uji Coba dengan Nilai C = 3 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	66.24	69.24	66.21	63.68	64.54
2	75.38	72.72	72.06	72.62	68.86
3	65.52	63.75	63.38	60.74	59.14
4	49.87	61.61	55.42	54.94	51.27
5	95.76	92.35	93.28	93.89	94.08
6	92.99	91.45	92.41	92.21	92.32
7	97.15	93.28	93.94	95.07	95.44
8	97.72	94.68	95.95	95.68	97.08
rata-rata	**80.08**	**79.88**	**79.08**	**78.60**	**77.84**

Tabel 7.41 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 3 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	69.85	54.93	65.91	70.84	72.87
2	76.06	60.83	72.11	75.75	75.22
3	68.71	52.64	68.38	74.01	78.08
4	85.03	54.77	69.22	76.24	79.19
5	79.58	98.62	94.25	89.15	86.73
6	94.04	91.54	93.69	96.50	97.22
7	83.15	99.07	94.54	89.75	86.02
8	74.10	93.12	86.79	84.39	82.86
rata-rata	**78.82**	**75.69**	**80.61**	**82.08**	**82.27**

Tabel 7.42 Hasil Akurasi (%) Uji Coba dengan Nilai C = 3 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	96.07	96.18	96.68	96.50	96.36
2	96.35	92.59	96.17	95.50	95.93
3	95.69	89.53	94.82	95.06	95.41
4	98.02	90.90	97.88	98.09	98.06
5	90.21	92.45	93.40	94.15	94.15
6	92.61	89.69	91.82	93.39	93.19
7	92.55	93.24	93.92	94.22	94.71
8	87.85	93.51	93.98	92.56	91.74
rata-rata	**93.67**	**92.26**	**94.83**	**94.93**	**94.94**

Tabel 7.43 Hasil Presisi (%) Uji coba dengan Nilai C = 3 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	92.85	95.54	94.24	93.28	93.12
2	94.64	96.68	95.23	93.44	93.99
3	95.48	97.54	96.26	96.47	96.55
4	97.89	99.77	98.37	98.37	98.11
5	99.73	92.44	94.20	97.12	98.32
6	99.32	89.69	93.30	96.60	98.18
7	99.89	93.32	94.96	96.67	98.70
8	99.72	93.47	95.54	96.86	97.60
rata-rata	**97.44**	**94.81**	**95.26**	**96.10**	**96.82**

Tabel 7.44 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 3 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	95.64	92.88	95.86	96.27	96.20
2	95.28	82.26	94.16	94.15	94.93
3	91.37	70.34	87.82	88.42	89.39
4	93.97	62.65	92.98	93.64	94.05
5	89.63	100.00	98.95	96.56	95.28
6	92.41	100.00	97.95	95.96	94.18
7	92.12	99.89	98.75	97.19	95.62
8	87.12	100.00	98.10	95.03	93.41
rata-rata	**92.19**	**88.50**	**95.57**	**94.65**	**94.13**

Tabel 7.45 Hasil Akurasi (%) Uji Coba dengan Nilai C = 4 untuk Metode *Yule-Walker.*

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	77.72	76.98	77.97	77.16	76.52
2	81.60	77.38	80.08	80.33	80.23
3	77.49	75.49	77.62	77.66	75.14
4	76.25	81.15	81.14	79.20	77.98
5	78.72	91.38	88.19	85.64	82.87
6	87.35	85.51	86.09	89.01	90.47
7	80.59	92.45	89.71	87.55	83.14
8	74.25	89.03	86.19	83.00	81.23
rata-rata	**79.25**	**83.67**	**83.37**	**82.44**	**80.95**

Tabel 7.46 Hasil Presisi (%) Uji Coba dengan Nilai C = 4 untuk Metode *Yule-Walker.*

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	65.63	69.65	67.30	64.83	63.21
2	73.83	72.69	71.77	71.76	70.67
3	65.66	66.11	65.51	64.33	60.25
4	50.82	63.53	59.75	55.54	53.00
5	96.12	92.38	93.28	93.45	93.69
6	93.19	91.40	92.25	92.58	92.89
7	96.44	93.26	93.77	94.91	95.27
8	97.35	94.52	96.31	96.23	96.24
rata-rata	**79.88**	**80.44**	**79.99**	**79.20**	**78.15**

Tabel 7.47 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 4 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	69.87	54.95	66.89	69.20	72.03
2	75.75	59.15	73.69	74.49	76.69
3	67.71	54.09	69.10	74.24	75.63
4	85.63	52.58	69.53	76.11	78.36
5	80.15	98.87	94.02	90.76	87.34
6	92.73	92.49	92.33	95.39	96.81
7	82.23	99.06	95.25	91.60	86.28
8	74.26	93.69	88.53	85.15	83.23
rata-rata	**78.54**	**75.61**	**81.17**	**82.12**	**82.05**

Tabel 7.48 Hasil Akurasi (%) Uji Coba dengan Nilai C = 4 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	rb RBF f (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	96.04	96.32	96.78	96.71	96.50
2	96.45	92.20	95.82	95.47	95.71
3	95.76	89.46	94.57	95.41	95.69
4	98.02	91.11	98.02	97.95	98.09
5	90.64	92.45	93.51	94.57	93.83
6	92.71	89.69	91.83	93.00	93.97
7	92.45	93.24	93.43	93.63	93.92
8	88.42	93.27	93.62	93.04	91.73
rata-rata	**93.81**	**92.22**	**94.70**	**94.97**	**94.93**

Tabel 7.49 Hasil Presisi (%) Uji Coba dengan Nilai C = 4 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	92.81	95.68	94.31	93.83	93.43
2	94.68	97.08	94.90	93.50	93.57
3	95.56	97.76	96.17	96.44	96.50
4	98.00	100.00	98.56	97.98	98.45
5	99.88	92.44	94.19	96.91	97.88
6	99.41	89.69	93.03	95.81	98.56
7	99.76	93.24	94.67	96.25	98.34
8	99.73	93.24	95.08	96.41	97.26
rata-rata	**97.48**	**94.89**	**95.11**	**95.89**	**96.75**

Tabel 7.50 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 4 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	rb RBF f (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	95.69	93.16	96.14	96.65	96.38
2	95.60	80.81	93.36	93.81	94.67
3	91.36	69.82	87.11	89.58	90.33
4	93.78	63.67	93.22	93.59	93.64
5	89.94	100.00	99.07	97.22	95.40
6	92.41	100.00	98.28	96.45	94.73
7	92.08	100.00	98.55	96.95	95.03
8	87.77	100.00	98.23	96.04	93.81
rata-rata	**92.33**	**88.43**	**95.49**	**95.04**	**94.25**

Tabel 7.51 Hasil Akurasi (%) Uji Coba dengan Nilai C = 5 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	78.68	77.19	77.69	77.30	77.37
2	81.60	77.94	82.40	81.31	79.84
3	77.94	76.33	78.08	77.31	76.29
4	77.44	81.25	80.42	79.02	78.30
5	79.26	91.70	88.62	85.43	83.51
6	87.06	84.73	85.03	88.52	89.78
7	80.39	92.55	90.59	87.06	85.29
8	75.80	89.73	86.90	83.01	82.78
rata-rata	**79.77**	**83.93**	**83.72**	**82.37**	**81.65**

Tabel 7.52 Hasil Presisi (%) Uji Coba dengan Nilai C = 5 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	66.94	70.44	66.76	65.04	64.31
2	73.81	73.92	75.42	72.89	69.88
3	66.00	67.87	66.39	63.67	61.60
4	52.15	64.22	58.41	55.06	53.80
5	95.87	92.39	92.96	93.67	93.69
6	93.52	92.00	93.02	92.90	92.09
7	96.59	93.19	94.01	94.96	95.18
8	97.29	94.41	96.04	96.54	97.01
rata-rata	**80.27**	**81.06**	**80.38**	**79.34**	**78.44**

Tabel 7.53 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 5 untuk Metode *Yule-Walker*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	71.74	54.57	65.71	69.20	72.10
2	75.89	59.64	75.97	76.38	77.28
3	69.26	54.44	69.07	73.16	77.57
4	84.62	51.76	68.99	75.28	77.47
5	80.99	99.20	94.79	90.55	87.98
6	91.99	90.93	90.02	94.40	96.94
7	81.77	99.27	95.96	90.93	88.77
8	76.17	94.53	89.58	84.76	83.97
rata-rata	**79.05**	**75.54**	**81.26**	**81.83**	**82.76**

Tabel 7.54 Hasil Akurasi (%) Uji Coba dengan Nilai C = 5 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	96.14	96.15	96.71	96.57	96.36
2	96.49	92.55	96.17	95.64	95.61
3	95.76	89.46	94.33	95.48	95.45
4	98.06	90.90	97.88	98.06	98.13
5	91.06	92.45	93.40	94.04	93.94
6	92.51	89.69	91.63	93.20	93.39
7	92.45	93.24	93.92	93.82	93.73
8	88.31	93.51	94.10	93.28	92.33
rata-rata	**93.85**	**92.24**	**94.77**	**95.01**	**94.86**

Tabel 7.55 Hasil Presisi (%) Uji Coba dengan Nilai C = 5 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	92.99	95.53	94.64	93.69	93.27
2	94.66	96.66	95.56	93.78	93.67
3	95.67	97.78	95.67	96.95	96.29
4	98.02	99.77	98.50	98.31	98.59
5	99.86	92.44	94.20	96.48	97.53
6	99.42	89.69	92.84	95.41	97.64
7	99.67	93.32	94.77	96.27	97.71
8	99.42	93.47	95.43	96.43	97.00
rata-rata	**97.46**	**94.83**	**95.20**	**95.92**	**96.46**

Tabel 7.56 Hasil Sensitivitas (%) Uji Coba dengan Nilai C = 5 untuk Metode *Burg*.

subjek	linear	RBF (γ =0.5)	RBF (γ =1)	RBF (γ =1.5)	RBF (γ =2)
1	95.73	92.75	95.54	96.18	95.94
2	95.57	82.15	93.76	94.24	94.12
3	91.43	69.89	86.87	89.27	89.90
4	93.76	62.65	92.82	93.63	93.57
5	90.42	100.00	98.95	97.14	95.84
6	92.21	100.00	98.27	97.09	94.91
7	92.21	99.89	98.97	97.17	95.46
8	87.85	100.00	98.36	96.32	94.66
rata-rata	**92.40**	**88.42**	**95.44**	**95.13**	**94.30**

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan dibahas mengenai kesimpulan yang dapat diambil dari tujuan pembuatan perangkat lunak, serta hasil uji coba yang telah dilakukan. Selain itu terdapat beberapa saran untuk pengembangan lebih lanjut.

6.1 Kesimpulan

Dari hasil pengamatan selama perancangan, implementasi, dan uji coba perangkat lunak, dapat diambil kesimpulan sebagai berikut:

1. Metode *autoregressive* (AR) model dan *support vector machine* (SVM) dapat dijadikan metode untuk ekstraksi fitur dan klasifikasi data EEG.
2. Metode *Burg* untuk mengestimasi koefisien AR model meghasilkan performa yang lebih baik daripada metode *Yule-Walker*.
3. Metode *Yule-Walker* menghasilkan rata-rata akurasi 80.60%, presisi 78.19%, dan sensitivitas 77.56%. Sedangkan metode *Burg* menghasilkan rata-rata akurasi 94.01%, presisi 95.70%, dan sensitivitas 93.39%.
4. Akurasi tertinggi metode *Yule-Walker* diperoleh ketika nilai *order* AR = 50, fungsi kernel menggunakan *Radian Basis Function* (RBF) dengan parameter *gamma* = 0.5 dan parameter C = 5.
5. Akurasi tertinggi metode *Burg* diperoleh ketika nilai *order* AR = 70, fungsi kernel menggunakan *Radian Basis Function* (RBF) dengan parameter *gamma* = 1 dan parameter C = 1.
6. Presisi tertinggi metode *Yule-Walker* diperoleh ketika nilai *order* AR = 50, fungsi kernel menggunakan *Radian Basis Function* (RBF) dengan parameter *gamma* = 0.5 dan parameter C = 5.

7. Presisi tertinggi metode *Burg* diperoleh ketika nilai *order* AR = 70, fungsi kernel menggunakan *linear*.
8. Sensitivitas tertinggi metode *Yule-Walker* diperoleh ketika nilai *order* AR = 50, fungsi kernel menggunakan *Radian Basis Function* (RBF) dengan parameter *gamma* = 1.5 dan parameter C = 1.
9. Sensitivitas tertinggi metode *Burg* diperoleh ketika nilai *order* AR = 70, fungsi kernel menggunakan Radian Basis Function (RBF) dengan parameter *gamma* = 1 dan parameter C = 3.

6.2 Saran

Berikut merupakan beberapa saran oleh penulis untuk pengembangan tugas akhir ini di masa yang akan datang, berdasarkan hasil perancangan, implementasi, dan uji coba yang telah dilakukan:

1. Menambahkan metode untuk menghilangkan *noise*, sehingga performa perangkat lunak dapat menjadi lebih baik.

DAFTAR PUSTAKA

[1] M. S. Scher, "Automated EEG-sleep analyses and neonatal neurointensive care," *Sleep Medicine,* vol. 5, pp. 533-540, 2004.

[2] M. Telpan, "Fundamental of EEG Measurement," *Measurement Science Review,* Vol. %1 dari %22, Section 2, 2002.

[3] A. Rechtschaffen dan A. Kales, A Manual of Standardized Terminology Techniques and Scoring System for Sleep Stages of Human Subjects, Los Angeles, USA: Brain Research Institute, UCLA, 1968.

[4] C. R. Burgess dan T. E. Scammell, "Narcolepsy: Neural Mechanisms of Sleepiness and Cataplexy," *The Journal of Neuroscience,* 2012.

[5] B. A. Shenoi, Introduction To Digital Signal Processing and Filter Design, New Jersey: Wiley Interscience, 2006.

[6] J. G. Proakis dan D. G. Manolakis, Digital Signal Processing, Upper Saddle River, New Jersey: Prentice Hall, 1996.

[7] M. H. hayes, Statistical Digital Signal Processing and Modeling, New York: John Wiley & Sons, 1996.

[8] P. Stoica dan R. L. Moses, Introduction of Spectral Analysis, Upper Saddle River, New Jersey: Prentice-Hall, 1997.

[9] G. J. McLachlan, K.-A. Do dan C. Ambroise, Analyzing Microarray Gene Expression Data, Wiley, 2004.

[10] A. S. Nugroho, A. B. Witarto dan D. Handoko, "Support Vector Machine-Teori dan Aplikasinya dalam Bioinformatika," *IlmuKomputer.Com,* 2003.

[11] T. Kayikcioglu, M. Maleki dan K. Eroglu, "Fast and Accurate PLS-Based Classification of EEG Sleep Using Single Channel Data," *Expert System with Applications,* pp. 7825-7830, 2015.

[12] K. Kazlauskas, "The Burg Algorithm with Extrapolation for Improving the Frequency Estimation," *INFORMATICA,* pp. 177-188, 2011.

[13] K. A. I. Aboalayon, H. T. Ocbagabir dan M. Faezipour, "Efficient Sleep Stage Classification Based on EEG Signals," *Instituite of Electrical and Electronics Engineers,* 2014.

BIODATA PENULIS

Yunan Helmi Mahendra merupakan anak kedua dari pasangan Bapak Makhin dan Ibu Endang Sri Purworini yang lahir di Mojokerto pada tanggal 9 Oktober 1994. Penulis menempuh pendidikan formal dimulai dari SDN Mentikan 4 Mojokerto (2000-2006), SMPN 2 Kota Mojokerto (2006-2009), SMAN 3 Kota Mojokerto (2009-2012), dan S1 Teknik Informatika ITS (2012-2016). Selama perkuliahan penulis pernah menjadi anggota organisasi Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS, Ikatan Mahasiswa Mojokerto ITS (IMMI), dan Keluarga Muslim Informatika (KMI) ITS. Di KMI penulis pernah menjabat sebagai Ketua Umum. Penulis juga mengikuti beberapa Latihan Keterampilan Manajemen Mahasiswa (LKMM) yaitu praTD dan TD. Selain aktif berorganisasi, penulis juga pernah mengikuti beberapa perlombaan dan menjadi finalis DevComp 2016. Penulis memilih bidang minat Komputasi Cerdas dan Visi (KCV) dan tertarik pada topik pengenalan pola dan kecerdasan komputasional. Penulis dapat dihubungi melalui email yunanhelmimahendra@gmail.com.