IMPLEMENTASI ALGORITME *EXTREME LEARNING MACHINE* (ELM) UNTUK KLASIFIKASI PENANGANAN *HUMAN PAPILLOMA VIRUS* (HPV)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Stefanus Bayu Waskito
NIM: 115060807111063

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI ALGORITME *EXTREME LEARNING MACHINE*(ELM) UNTUK KLASIFIKASI PENANGANAN *HUMAN PAPILLOMA VIRUS*(HPV)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :
Stefanus Bayu Waskito
NIM: 115060807111063

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin S.Si, M.Kom	Edy Santoso S.Si, M.Kom
NIK. 201201 850719 1 001	NIP. 19740414 200312 1 004

a.n. Imam Cholissodin

Mengetahui
Ketua Jurusan Teknik Informatika

Wayan Ast oro Kurniawan S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

ii

IDENTITAS TIM PENGUJI

Penguji 1 : Ratih Kartika Dewi, S.T., M.Kom

NIK. 201503 890520 2 001

Penguji 2 : Nurudin Santoso, S.T., M.T

NIP. 19740916 200012 1 001

PERNYATAAN ORISINALITAS

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

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

Malang, 3 Agustus 2018

Stefanus Bayu Waskito

NIM: 115060807111063

DAFTAR RIWAYAT HIDUP

Nama	: Stefanus Bayu Waskito
Tempat/Tanggal Lahir	: Kediri/9 September 1993
Jenis Kelamin	: Laki-laki
Agama	: Katolik
Riwayat Pendidikan	: - SD Angkasa 2 Medan
	- SMP Negeri 2 Medan
	- SMA Negeri 13 Medan

KATA PENGANTAR

Puji syukur penulis panjatkan Tuhan Yang Maha Esa karena karena penulis dapat menyelesaikan skripsi dengan judul "Implementasi Algoritme Extreme Machine Learning untuk Klasifikasi Penanganan *Human Papilloma Virus*". Penulisan skripsi ini disusun untuk memenuhi syarat menjadi Sarjana Komputer.

Dalam pelaksanaan dan penulisan skripsi ini penulis mendapatkan banyak bantuan dari berbagai pihak baik secara moril dan materiil. Dalam kesempatan ini penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Imam Cholissodin, S.Si., M.Kom., selaku pembimbing utama dan Edy Santoso, S.Si., M.Kom., selaku pembimbing pendamping yang telah meluangkan waktu memberikan pengarahan dan bimbingan kepada penulis.
2. Tri Astoto Kurniawan, S.T, M.T, Ph.D., selaku Ketua Jurusan Teknik Informartika Fakultas Ilmu Komputer.
3. Ir. Heru Nurwasito, M.Kom, selaku Wakil Ketua I Bidang Akademik Fakultas Ilmu Komputer.
4. Agus Wahyu Widodo, S.T., M.Cs, selaku Ketua Program Studi Teknik Informatika yang selaku memberikan memberikan dukungan kepada penulis baik berupa doa maupun hal yang bersifat administratif.
5. Segenap dosen Fakultas Ilmu Komputer Universitas Brawijaya atas segenap ilmu pengetahuan dan perhatian yang diberikan.
6. Keluarga Agus Sudaryono yang telah menjadi panutan dan inspirasi semangat bagi penulis, MG Anik Endrawati yang selalu memberikan dukungan dan doa yang tiada henti.
7. Seluruh teman-teman Teknik Informatika angkatan 2011 atas seluruh dukungan dan kebersamaannya dari awal perkuliahan sampai akhir.
8. Serta semua pihak yang telah membantu dan memberikan pengalaman berharga bagi penulis selama penulis menjalani masa perkuliahan.

Penulis sadar bahwa skripsi ini masih jauh dari kesempurnaan, untuk itu dengan segala kerendahan hati penulis mengharapkan saran dan kritik yang membangun demi kesempurnaan penulisan selanjutnya.

Malang, Juli 2018

Stefanus Bayu Waskito

kitowas@gmail.com

ABSTRAK

Stefanus Bayu Waskito, Implementasi Algoritme *Extreme Learning Machine* (ELM) untuk Klasifikasi Penanganan *Human Papilloma Virus* (HPV)

Pembimbing : Iman Cholissodin S.Si, M.Kom dan Edy Santoso S.Si, M.Kom

Human Papilloma Virus merupakan virus yaxng umumnya menyebabkan kutil dan mata ikan. *Human Papilloma Virus* memiliki metode penanganan yang cukup banyak namun penanganan dengan menggunakan *Imunnotherapy* dan *Cryotherapy*. Didasarkan dari banyaknya metode penanganan *Human Papilloma Virus* makan dilakukan penelitian guna mengklasifikasikan metode penanganan *Human Papilloma Virus* yang paling tepat berdasarkan parameter gejala yang ada. Pada penelitian dengan menggunakan metode *Extreme Learning Machine* untuk klasifikasi metode penanganan *Human Papilloma Virus* dilakukan pengujian untuk mengetahuui pengaruh fungsi aktivasi , jumlah *hidden neuron* dan rasio data terhadap akurasi dari hasil klasifikas. Selain itu juga dilakukan pengujian terhadap pengaruh jumlah *hidden neuron* terhadap lama waktu proses klasifikasi. Berdasarkan dari hasil pengujian yang dilakukan, akurasi yang didapatkan sistem dalam klasifikasi metode penanganan *Human Papilloma Virus* memiliki akurasi yang baik dengan akurasi sebesar 70,8% dengan menggunakan fungsi aktivasi *Sigmoid Biner*, rasio data latih uji 80:20 dan *hidden neuron 10 buah.* Selain itu waktu yang diperlukan untuk melakukan klasifikasi berdasarkan parameter terbaik memiliki waktu yang cukup cepat dengan waktu selama 0,043 detik.

Kata kunci: *human papilloma virus, klasifikasi, cryotherapy, imunnotherapy, extreme learning machine*

ABSTRACT

Stefanus Bayu Waskito, Extreme Learning Machine Algorithm Implementation for Classify Human Papilloma Virus Treatment Method

Advisor : Imam Cholissodin S.Si., M.Kom. and Edy Santoso S.Si., M.Kom

Human Papilloma is a virus that cause warts ilness. There are several treatment methods, but Immunotherapy and Cryotherapy are considered to be the best method to treat this ilness. However, none of them can heal all patients. Therefore, research to determine which method more appropriate for a certain patient is required. This research use Extreme Learning Machine Algorithm to help classify which method are better for certain patient. A tests is conducted to determine the effects of activation function, number of hidden neuron and and data ratio toward classification accuracy. It was observed that using Binary Sigmoid activation function, 80 testing data to 20 training data ratio, and 10 hidden neuron, the classification accuraccy reach 70,8%. And the classification time spent were relatively fast that is only 0.043 seconds.

Keywords: *human papilloma virus, classification, cryotherapy, imunnotherapy, extreme learning machine*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

Bagian utama skripsi terdiri dari beberapa komponen atau bab yang tersusun dengan alur yang logis. Pendahuluan merupakan komponen/bab pertama yang harus menjelaskan apa yang dikerjakan dalam skripsi dan mengapa ini dikerjakan.

1.1 Latar belakang

Human Papilloma Virus(HPV) merupakan virus yang umumnya menjangkit manusia pada rentang usia antara 1 sampai dengan 30 tahun. Penyakit yang disebabkan oleh *Human Papilloma Virus*(HPV) adalah kutil, mata ikan dan bahkan dapat menyebabkan kanker serviks, namun kutil(*veruca plantaris*) dan mata ikan(*veruca vulgaris*) merupakan penyakit yang sering ditemu yang dapat digolongkan sebagai tumor jinak (Khozeimeh, et al., 2017).

Menurut Azad pada prakteknya dokter maupun praktisi kesehatan memerlukan waktu dalam pengambilan keputusan dalam mengklasifikasikan metode penanganan yang tepat pada pasien yang terinfeksi *Human Papilloma Virus*(HPV). Proses yang lama dalam pengambilan keputusan untuk pemilihan penanganan *Human Papilloma Virus*(HPV) umumnya disebabkan oleh banyaknya parameter yang harus diperiksa dan banyaknya metode penanganan untuk menanganani infeksi *Human Papilloma Virus*(HPV). Metode penanganan yang paling umum digunakan untuk penyembugan kutil dan mata ikan yang disebabkan oleh infeksi virus *Human Papilloma Virus*(HPV) adalah dengan melakukan operasi pengangkatan, pemberian *bleyomycin*, *Imunnotherapy*, dan *Cryotherapy.* Penanganan menggunakan *Imunnotherapy* dan *Cryotherapy* merupakan metode penanganan yang terbaik (Azad, et al., 2017).

Beberapa penelitian mengenai klasifikasi terhadap permasalan ada sebelumnya pernah dilakukan oleh Fahime mengenai pemilihan metode penanganan *Human Papilloma Virus*(HPV) dengan metode *fuzzy* Sugeno dengan akurasi sebesar 83,33% (Khozeimeh, et al., 2017). Makrina Christy yang mengangkat penelitian klasifikasi penyimpangan tumbuh kembang anak menggunakan metode ELM. Pada penelitian tersebut Makrina menyebutkan bahwa metode ELM menghasilkan akurasi sistem sebesar 76,67% dengan menggunakan fungsi aktivasi Sigmoid Biner (Christy, et al., 2017).

Pada penelitian lain oleh Ersa menyebutkan bahwa algoritme ELM mendapatkan akurasi sebesar 78,94% dengan lama waktu proses 5,71 detik pada penelitian berjudul klasifikasi kondisi mata berdasarkan sinyal EEG dengan menggunakan ELM (Prakoso, et al., 2016). Berdasarkan penelitian lain yang dilakuakn oleh Siti Aliza yang berjudul klasifikasi aritmia berdasarkan sinyal EKG dengan menggunakan ELM menyebutkan bahwa algoritme ELM yang diterapkan mendapatkan akurasi sebesar 100% dengan *learning rate* selama 0,35 detik (Nur, et al., 2011).

Berdasarkan permasalahan yang ada dan hasil penelitian sebelumnya yang menyatakan bahwa algoritme ELM memliki tingkat akurasi yang tinggi dan *learning rate* yang cepat maka penulis mengangkat penelitian yang bejudul " Implementasi Algoritme *Extreme Learning Machine*(ELM) untuk Klasifikasi Penanganan *Human Papilloma Virus*(HPV).

1.2 Rumusan masalah

Berdasarkan latar belakang dan permasalahan yang ada penulis menyimpulkan rumusan masalah yang diangkat sebagai berikut:

1. Bagaimana menerapkan algoritme *Extreme Learning Machine*(ELM) untuk mengklasifikasikan penanganan *Human Papilloma Virus*(HPV)?
2. Bagaimana tingkat akurasi algoritme *Extreme Learning Machine*(ELM) pada klasifikasi penanganan *Human Papilloma Virus*(HPV)?

1.3 Tujuan

Berdasarkan rumusan masalah yang ada, tujuan yang dapat diambil pada penelitian ini adalah sebagai berikut:

1. Menerapkan algoritme *Extreme Learning Machine*(ELM) untuk klasifikasi penanganan *Human Papilloma Virus*(HPV)
2. Menguji tingkat akurasi algortime *Extreme Learning Machine*(ELM) pada klasifikasi penanganan *Human Papilloma Virus*(HPV)

1.4 Manfaat

1. Mendapatkan wawasan mengenai penerapan algoritme *Extreme Learning Machine*(ELM) untuk klasifikasi penanganan *Human Papilloma Virus*(HPV)
2. Membantu dalam mempermudah dan mempercepat dokter maupun praktisi kesehatan dalam proses pemilihan metode penanganan yang tepat untuk menangani penyakit yang disebabkan oleh *Human Papilloma Virus*(HPV)

1.5 Batasan masalah

Batasan maslah yang dikemukakan pada penelitian ini bertujuan untuk membatasi masalah dalam ruang lingkup penelitian yang dilakukan. Batasan masalah yang diajukan adalah sebagai berikut:

1. Data yang digunakan berjumlah 180 data yang berasal dari repositori data University of California Irvine
2. Data yang digunakan merupakan rujukan dari penelitian Fahime Khoziemeh dan rekan yang berjudul *An Expert System for Selecting Wart Treatment Method* yang dipublikasi pada tahun 2017
3. Metode penanganan *Human Papilloma Virus*(HPV) yang diajukan hanya ada dua metode yaitu *Imunnotherapy* dan *Cryotherapy* yang dirujuk dari peneitian yang dilakukan Fahime Khoziemeh

4. Algoritme yang digunakan pada penelitian ini adalah algoritme *Extreme Learning Machine*(ELM)

1.6 Sistematika pembahasan

Sistematika penulisan pada penelitian ini digunakan untuk memahami sistematika pembahasan yang dituliskan pada laporan penelitian. Berikut ini merupakan garis besar pembahasan pada penelitian yang dilakukan.

BAB I Pendahuluan

Pada bab pendahuluan akan diuraikan permasalahan yang mendasari penelitian ini. Selain dari permasalahan, rumusan masalah, tujuan dan manfaat juga disampaikan pada penelitian ini.

BAB II Landasan Kepustakaan

Pada bab landasan kepustakaan akan diuraikan dasar teori yang berhubungan dengan penelitian ini. Selain itu, terdapat juga kajian pustaka terhadap penelitian yang menggunakan algoritme maupun objek yang sama yang digunakan sebagai pembanding maupun sebagai rujukan pada penelitian ini.

Bab III Metodologi Penelitian

Pada bab metodologi penelitian akan diuraikan metodologi yang digunakan pada penelitian ini berupa langkah yang akan dilakukan peneliti dalam menyelesaikan permasalahan dari masalah yang didapat pada pendahuluan.

BAB IV Perancangan

Pada bab perancangan akan diuraikan rancangan model, rancangan antarmuka sistem, dan rancangan pengujian yang akan diimplementasikan pada penelitian ini.

BAB V Implementasi

Pada bab implementasi akan diuraikan penerapan dari rancangan yang sudah dibuat pada bab perancangan menggunakan bahasa pemrograman yang telah diajukan. Implementasi yang akan dilakukan meliputi implementasi antarmuka dan implementasi kode program.

BAB VI Pengujian dan Analisis

Pada bab pengujian dan analisis akan diuraikan teknis pengujian yang dilakukan dan selain itu akan diuraikan pula hasil analisis dari hasil pengujian yang didapatkan.

BAB VII Penutup

Pada bab penutup akan diuraikan kesimpulan yang telah didapatkan dari implementasi, pengujian, dan analisis hasil pengujian terhadap permasalahan yang diangkat. Selain itu akan diuraikan pula saran yang diambil dari kekurangan sistem dan usulan pengembangan penelitian untuk kedepannya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Pada kajian pustaka yang dibahas pada sub bab ini mengenai penelitian terdahulu yang memiliki kesamaan objek maupun metode yang digunakan pada penelitian ini. Berikut ini merupakan penelitian terdahulu yang dimaksudkan.

Tabel 2.1 Kajian Pustaka

No	Judul	Objek Parameter masukan	Metode	Hasil
1	*An Expert System for Selecting Wart Treatment Method*	*Human Papilloma Virus* Jenis kelamin, usia, jama terjangkit, tipe, luas area terjangkit, respon terhadap penanganan	FIS Sugeno, FIS Mamdani	Akurasi yang didapatkan pada *Imunnotherapy* sebesar 83,33% dan *Cryotherapy* sebesar 80,7%
2	Klasifikasi Penyimpangan Tumbuh Kembang Anak Menggunakan Metode *Extreme Learning machine*	Parameter tumbuh kembang anak, dan 100 data tumbuh kembang anak	*Extreme Learning Machine*	Metode ELM dapat diterapkan, akurasi yang didapat sebesar 76,67% dengan penggunaan fungsi aktivasi Sigmoid Biner dan penggunaan 10 *hidden neuron*
3	Klasifikasi Keadaan Mata Berdasarkan Sinyal EEG dengan Menggunakan *Extreme Learning Machine*	Keadaan Mata Data set keadaan mata	*Extreme Learning Machine*	Metode ELM dapat diterapkan, akurasi yang didapat sebesar 78,94% dan *learning rate* 5,71 detik
4	Klasifikasi Aritmia pada Sinyal EKG Menggunakan *Extreme Learning Machine*	Aritmia Jantung Data set EKG jantung	*Extreme Learning Machine*	Metode ELM dapat diterapkan, akurasi yang didapat sebesar 100% dengan *learning rate* 0,35 detik

(Khozeimeh, et al., 2017), (Christy, et al., 2017), (Prakoso, et al., 2016), (Nur, et al., 2011)

Berdasarkan penelitian sebelumnya yang dilakukan oleh Nur yang berjudul "Klasifikasi Aritmia pada Sinyal EKG menggunakan *Extreme Learning Machine*" menyebutkan bahwa tingkat akurasi pada permasalahan klasifikasi ini sebesar 100% dengan waktu proses selama 0,35 detik. Pada penelitian ini nur menyatakan bahwa semakin banyak *hidden node* berpengaruh pada tingkat akurasi dari sistem namun *learning rate* yang didapatkan akan semakin besar. Selain dari pengaruh *hidden node,* nilai akurasi 100% yang didapatkan menggambarkan bahwa algoritme *Extreme Learning Machine* dapat mengenali seluruh data uji dikarenakan proses latih yang dilakukan berjalan dengan tepat dan persebaran data yang digunakan berimbang (Nur, et al., 2011).

2.2 *Human Papilloma Virus*

Human Papilloma Virus merupakan virus yang menyebabkan kutil yang mana dapat dikategorikan sebagai tumor jinak. Pada umumnya kutil ditemukan pada permukaan kulit tangan dan kaki. Kutil yang disebabkan oleh HPV selain menginfeksi permukaan kulit tangan maupun kaki ada juga yang menyebabkan kutil di sekitar perukaan kulit kelamin seperti penis, vagina maupun dubur. Biasanya HPV menjangkit orang yang berusia antara 1 sampai 30 tahun namun tidak menutup kemungkinan menjangkit segala usia. (Moore, 1998)

Menurut klinik dermatologi rumah sakit Ghaem di Mashhad Iran, dampak yang paling sering diderita olah pasien akibat HPV adalah kutil (*Common Warts*) dan mata ikan (*Plantar Warts*). Banyak cara penangnan HPV dalam hal ini kutil (*Common Warts*) dan mata ikan (*Plantar Warts*) diantaranya *cryotherapy*, *immunotherapy*, pembedahan, ineksi bleomycin dan banyak jenis penanganan lainnya. Namun menurut penelitian Fahime, *immunotherapy* dan *cryotherapy* merupakan metode penanganan HPV yang dianggap paling baik (Khozeimeh, et al., 2017).

2.2.1 *Imunnotherapy*

Imunotherapy merupakan sebuah terapi biologis yang digunakan untuk merangsang kekebalan tubuh guna membantu melawan kanker, infeksi maupun penyakit lainnya dan dapat digunakan untuk penangan *Human Papilloma Virus* (Thappa & Chiramel, 2016). Berdasarkan penelitian yang dilakukan oleh Clifton menyatakan bahwa 47% pasien yang terinfeksi *Human Papilloma Virus* dapat disembuhkan menggunakan *Imunnotherapy.*

Immunotherapy merupakan metode pengembangan dari penanganan *Human Papilloma Virus* yang mana antigen *Candida* akan diinjeksi langsung ke are kutil maupun mata ikan. Penanganan infeksi *Human Papilloma Virus* menggunakan *Imunnotherapy* dengan injeksi antigen *Candida* dinilai memberi efek rasa sakit dibanding dengan metode lainnya seperti *Cryotherapy* (Clifty, 2003).

2.2.2 *Cryotherapy*

Cryotherapy merupakan salah satu metode penyembuhan HPV, metode ini sering disebut juga *freezing treatment.* Pada metode penanganan HPV menggunakan metode ini, nitrogen cair akan didiberikan ke area yang terinfeksi. Pemberian nitrogen cair untuk penanganan *Human Papilloma Virus* dilakukan berdasarkan penelitian sebelumnya oleh Bunney pada tahun 1970. Pemberian nitrogen cair pada area yang terinfeksi *Human Papilloma Virus* menimbulkan efek samping rasa sakit pada penderita (Bourke, Berth-Jones, & Hutchinson, 1995).

Pelaksanaan penanganan menggunakan *Cryotherapy* umumnya dilakukan satu minggu sekali selama satu sampai tiga minggu namun ada juga yang melakukan *Cryotherapy* sampai tiga bulan tergantung sudah berapa lama terinfeksi dan ukuran kutil maupun mata ikan yang diderita oleh pasien. Berdasarkan penelitian Khozimeh menyatakan bahwa 80,7% penderita dapat disembuhkan menggunakan metode ini (Khozeimeh, et al., 2017).

2.3 Normalisasi

Normalisasi menurut Patro adalah proses yang dilakukan untuk mendapatkan data dengan skala tertentu yang mewakili rentang data asli itu sendiri. Pada penelitian ini menggunakan Min-Max normalisasi sesuai dengan persamaan 2.1. Min-Max normalisasi merupakan metode normalisasi dengan melakukan transformasi linier terhadap data asli. Keunggulan dari Min-Max normalisasi adalah memiliki keseimbangan nilai perbandingan antara data saat sebelum dan sesudah proses normalisasi. Berikut ini merupakan persamaan yang digunakan untuk melakukan proses normalisasi (Patro & Kishore, 2015).

$$y = \frac{x-min}{max-min} * (B_{atas} - B_{bawah}) + B_{bawah} \quad (2.1)$$

Dengan nilai x merupakan nilai dari data yang akan dinormalisasi, min merupakan nilai data minimum dan max merupakan nilai data maksimum. B_{atas} merupakan batas atas dan B_{bawah}merupakan batas bawah dari rentang nilai normalisasi. Pada penelitian ini batas atas yang digunakan adalah 0,99 dan batas bawah memiliki nilai 0,1.

Selain normalisasi data ada juga denormalisasi data. Denormalisasi merupakan proses yang dilakuakn untuk mendapatkan nilai asli dengan cara mengembalikan ukuran data yang telah dinormalisasi sebelumnya. Berikut ini merupakan persamaan yang digunakan untuk melakukan denormalisasi data.

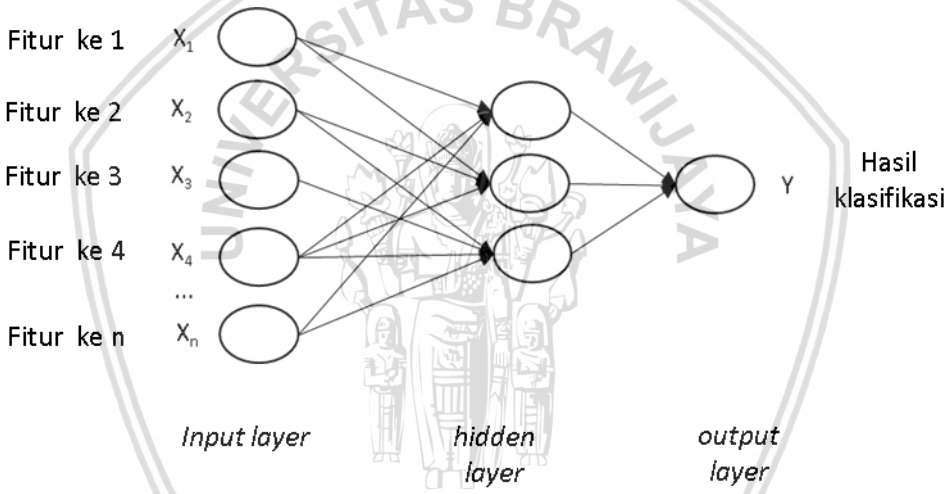
$$x = \frac{y-B_{bawah}}{B_{atas}-B_{bawah}} * (max - min) + min \quad (2.2)$$

Pada persamaan 2.2 dapat dijabarkan bahwa y merupakan nilai normalisasi yang akan di denormalisasi, max merupakan nilai data maksimum, min merupakan nilai data minimum. Selain nilai max dan min terdapat juga B_{atas} merupakan batas atas dan B_{bawah} yang merupakan batas bawah dari *range* nilai normalisasi.

2.4 *Extreme Learning Machine*

Extreme Learning Machine merupakan sebuah skema pembelajaran data yang ada pada jaringan syaraf tiruan yang pertama kali dibahas pada konferensi mengenai jaringan syaraf tiruan yang dilaksanakan di Budepaest Hungaria pada tahun 2004. Algoritme *Extrema Learning machine* yang digagas oleh Huang memiliki proses pembelajaran yang jauh lebih cepat dibandingkan dengan algoritma jaringan syaraf tiusan lain yang berbasis gradien (Zhu, et al., 2005).

Pada algoritme ini bobot masukan dan bias yang digunakan akan dipilih secara acak berdasarkan rentang tertentu bukan seperti algoritme jaringan syaraf lain yang bobot masukan dan biasnya dipilih atau ditentukan satu per satu. Pengujian performa dilakukan oleh Huang untuk mengetahui performa *Extreme Learning Machine* pada klasifikasi dan prediksi dan dapat diambil kesimpulan bahwa algoritme *Extreme Learning Machine* menghasilkan performa yang lebih baik dan waktu proses yang lebih cepat dibandingkan dengan metode jaringan syaraf tiruan lainnya (Huang, et al., 2004).



Gambar 2.1 Arsitektur *Extreme Learning Machine*

Arsitektur *Extreme Learning Machine* yang terdapat pada Gambar 2.1 menunjukkan bahwa terdapat *input layer*, *hidden layer*, dan *output layer*. Pada *input layer* nantinya terdapat masukan sejumlah fitur yang digunakan mulai dari fitur pertama sampai fitur ke *k*. Setelah *input layer*, nilai masukan akan dimasukkan pada matriks W_{jk}sebagai bobot masukan yang nilainya dipilih secara acak. Bobot masukan yang ada nantinya akan digunakan untuk perhitungan nilai *H* pada *hidden neuron* menggunakan fungsi aktivasi. Setelah dilakukan perhitungan nilai *H* nantinya akan menghasilkan nilai $\hat{\beta}$ yang digunakan sebagai nilai masukan *output layer* untuk menghitung nilai keluaran $\hat{Y}_i$.

2.4.1 Fungsi Aktivasi

Srimuang dan Intarasothonchun menyebutkan ada 7 fungsi aktivasi yang digunakan pada algoritme *Extreme Learning Machine.* Ketujuh fungsi aktivasi yang ada pada algoritme *Extreme Learning Machine* adalah sebagai berikut:

1. Fungsi Sigmoid Biner

$$H = \frac{1}{1+\exp(-H_{init})} \quad (2.3)$$

2. Fungsi Linear

$$H = H_{init} \quad (2.4)$$

3. Fungsi Sin

$$H = \sin(H_{init}) \quad (2.5)$$

4. Fungsi Radial Basis

$$H = \exp(-(H_{init})^2) \quad (2.6)$$

5. Fungsi Sigmoid Bipolar

$$H = \frac{1-\exp(-H_{init})}{1+\exp(-H_{init})} \quad (2.7)$$

6. Fungsi Hard Limit

$$H = \begin{cases} 1, jika\ Hinit \geq 0 \\ 0, jika\ tidak \end{cases} \quad (2.8)$$

7. Fungsi Triangular Basis

$$H = \begin{cases} 1 - abs\ H_{init}, jika - 1 \leq H_{init} \leq 1 \\ 0, jika\ tidak \end{cases} \quad (2.9)$$

2.4.2 Proses Latih *Extreme Learning Machine*

Pada algoritme *Extreme Learning Machine* proses latih akan dilakukan melalui langkah-langkah sebagai berikut:

1. Nilai bobot masukan dan nilai bias bidapat secara acak.
2. Menghitung nilai keluaran matriks *hidden layer.*

$$H = \frac{1}{1+\exp(-X_{training}.W^T + ones(N_{train},1) * bias)} \quad (2.10)$$

X merupakan nilai matriks data latih dan W^T adalah nilai *transpose* dari matriks bobot masukan.

3. Menghitung nilai matriks *Moore-Penrose*

$$H^+ = (H^T.H)^{-1}.H^T \quad (2.11)$$

H^+ merupakan nilai matriks *Moore-Penrose* dari matriks H, matriks H merupakan matriks yang tersusun dari keluaran masing-masing *hidden layer* dan Y merupakan matriks target.

4. Menghitung nilai matriks bobot keluaran dari *hidden layer* .

$$\hat{\beta} = H^{+}\, Y \tag{2.12}$$

$\hat{\beta}$ merupakan matriks bobot keluaran, H^{+} merupakan matriks *Moore-Penrose* dan *Y* merupakan matriks target.

2.4.3 Proses Uji *Extreme Learning Machine*

Pada algoritme *Extreme Learning Machine* proses uji akan dilakukan melalui langkah-langkah sebagai berikut:

1. Menghitung nilai matriks keluaran *hidden layer.*

$$H = \frac{1}{1+\exp(-X_{testing}.W^{T} + ones(N_{test},1) * bias)} \tag{2.13}$$

X merupakan matriks data uji dan W^{T} merupakan *transpose* matriks bobot.

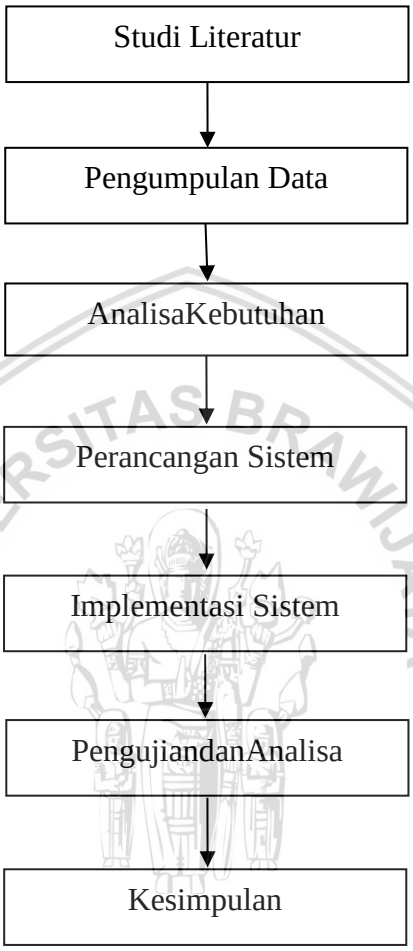
2. Menghitung nilai Y prediksi.

$$\hat{Y} = H.\hat{\beta} \tag{2.14}$$

$\hat{Y}$ merupakan matriks target prediksi, H merupakan matriks keluaran *hidden layer* dan $\hat{\beta}$ merupakan matriks bobot keluaran yang diperoleh dari proses latih.

BAB 3 METODOLOGI

Metodologi penelitian yang dilakukan dalam penelitian ini melalui beberapa tahapan-tahapan yang dapat diilustrasikan dengan diagram blok metodologi penelitian seperti pada Gambar 3.1:



Gambar 3.1 Blok Diagram Metodologi Penelitian

3.1 Studi Literatur

Tahapan studi literatur dilakukan bertujuan untuk mendapatkan informasi yang nantinya akan digunakan sebagai acuan dalam pelakasanaan penelitian ini. Literatur yang diperlukan sebagai acuan pada penelitian algoritme *Extreme Learning Machine* untuk klasifikasi penanganan *Human Papilloma Virus* diantaranya :

- *Human Papilloma Virus*
- Normalisasi data
- *Extreme Machine Learning*

Literatur yang diperlukan untuk menunjang penelitian ini diperoleh dari buku, jurnal, artikel dan dokumentasi lainnya.

3.2 Pengumpulan Data

Pada penelitian mengenai klasifikasi penaganan *Human Papilloma Virus* menggunakan algoritme *Extreme Learning Machine*, data yang digunakan berasal dari UCI yang merupakan data penelitian yang dilakukan oleh Fahime yang berjudul *An Expert Syetem for Selecting Wart Treatment Method.* Penelitian tersebut dilaksanakan pada tahun 2017 dan di *publish* pada 4 Januari 2018. Berdasarkan dari penelitian Fahime, data yang digunakan berjumlah 118 data dengan 7 fitur yang ada.

3.3 Perancangan Sistem

Tahapan perancangan sistem dilakukan setelah tahapan analisis kebutuhan baik kebutuhan perangkat lunak maupun perangkat keras telah terpenuhi. Pada implementasi algoritme *Extreme Learning Machine* untuk klasifikasi penanganan *Human Papilloma Virus* telah dirancang menjadi beberapa tahapan. Tahapan pertama yang dilakuakn pada penelitian ini adalah melaksanakan normalisasi data yang telah ada, baik itu data latih maupun data uji. Setalah tahapan normalisasi data dilakukan tahap latih data terhadap algoritme *Extreme Learning Machine* kemudian tahapan terakhir merupakan tahapan uji.

3.4 Implementasi

Setelah melakukan tahapan perancangan sistem , tahapan yang dilakukan selanjutnya adalah melakukan implementasi. Implementasi yang dilakukan pada penelitian ini adalah dengan membuat program aplikasi menggunakan bahasa pemrograman C# pada Microsoft Visual Studio 2010 dan selain itu juga melakukan implementasi perhitungan manua algoritme *Extreme Learning Machine* pada Microsoft Excel 2016.

3.5 Pengujian dan Analisis

Tahapan yang dilakukan setelah tahapan implementasi adalah tahapan pengujian terhadap program aplikasi maupun manualisasi. Pada tahapan pengujian program aplikasi, pengujian dibagi menjadi 4 pengujian antara lain pengujian terhadap pengaruh rasio perbandingan data latih dan data uji terhadap akurasi, pengujian terhadap pengaruh jumlah *hidden neuron* terhadap akurasi, pengujian pengaruh fungsi aktivasi terhadap akurasi dan pengujian pengaruh jumlah *hidden neuron* terhadap lama waktu proses.

Setelah tahapan pengujian dilanjutkan ke tahapan analisis hasil pengujian. Analisis dilakukan untuk proses pengambilan kesimpulan terhadap hasil pengujian yang telah dilakukan. Data hasil pengujian yang dianalisis adalah data akurasi maupun lama waktu proses dalam klasifikasi menggunakan *Extreme Learning Machine.*

3.6 Kesimpulan

Tahapan pengambilan kesimpulan dilakukan setaleh tahapan uji dan analisis terhadap hasil uji sistem maupun analisis algoritme yang telah diimplementasikan. Tahapan terakhir yang dilakukan setelah penulisan kesimpulan adalah tahapan penulisan saran. Penulisan saran dimaksudkan agar memberi gambaran untuk penelitian pengembangan terhadap penelitian ini.

BAB 4 PERANCANGAN

Pada bab perancangan akan dilakukan pembahasan mengenai rancangan sistem yang akan dibuat sebagai langkah implementasi metode yang digunakan pada penelitian ini. Perancangan yang dilakukan diawali dengan identifikasi masalah, perancangan antarmuka sistem, perancangan algoritme *Extreme Learning Machine,* perhitungan manual, dan perancangan skenario pengujian sistem.

4.1 Identifikasi Permasalahan

Pada penelitian ini permasalahan yang harus diselesaikan adalah melakuakan klasifikasi penanganan *Human Papilloma Virus* dengan menggunakan algoritme *Extreme Learning Machine*. Data masukan dari pengguna nantinya akan diolah dengan menggunakan algoritme *Extreme Learning Machine* untuk mengetahui prediksi penanganan yang tepat terhadap *Human Papilloma Virus.* Sampel data yang digunakan berjumlah 180 data menengnai *Human Papilloma Virus* sebagaimana yang ditunjukkan pada Tabel 4.1 berikut. (sampel data yang lebih lengkap akan ditunjukkan pada lampiran)

Tabel 4.1 Sampel Data

No.	Sex	Age	Time	Number of Warts	Type	Area	Result of Treatment
1	1	19	8	8	1	160	1
2	1	16	11	3	2	60	1
3	1	24	9,5	8	1	20	1
4	1	35	9,25	9	1	100	1
5	2	22	5	9	1	70	2
6	2	15	1,5	12	3	70	2
7	2	17	5,25	3	1	63	2
8	2	15	6	2	1	30	2
9	1	29	8,75	3	1	504	1
10	1	27	5	2	1	20	1
11	1	16	10,5	2	1	100	1
12	1	29	5	12	3	75	1
13	2	51	4	1	1	65	2
14	2	36	1,75	10	3	45	2
15	2	53	7,25	6	1	81	2
16	2	19	6	2	1	225	2
17	1	63	2,75	3	3	20	1

18	1	67	3,75	11	3	20	1
19	1	23	10,25	7	3	72	1
20	1	36	11	2	1	8	1
21	2	32	12	12	3	750	2
22	2	32	12	4	3	750	2
23	2	34	12	3	3	95	2
24	2	23	9,5	5	3	72	2

Klasifikasi penanganan *Human Papilloma Virus* ini menggunakan 6 fitur yaitu *Sex, Age, Time,Number of Warts, Type, Area* dan *Result of treatment* seperti yang tertera pada Tabel 4.1. Enam fitur tersebut menjadi jumlah *input layer* yang nantinya menjadi salah satu parameter dalam proses *Extreme Learning Machine*. Sedangkan *Result of Treatmrent* menjadi target yang juga merupakan parameter dalam proses algoritme *Extreme Learning Machine*.

Parameter yang akan digunakan pada proses *Extreme Learning Machine* meliputi: jumlah *hidden neuron*, matriks bobot *W*, matriks bias, jumlah *input layer*, jumlah *target*, jumlah data latih dan data uji. Penentuan nilai matriks bobot *W* dilakukan secara acak dengan rentang nilai antara -1 sampai 1. Matriks bobot *W* mempunyai *ordo* jumlah *hidden neuron* x jumlah *input layer*. Untuk matriks bias mempunyai *ordo*1 x jumlah *hidden neuron* dan mempunyai rentang nilai antara 0 sampai 1 secara acak. Parameter-parameter tersebut digunakan untuk fase latih, fase ini akan menghasilkan matriks bobot keluaran $\hat{\beta}$ yang digunakan pada fase uji untuk menghitung matriks prediksi $\hat{Y}$.

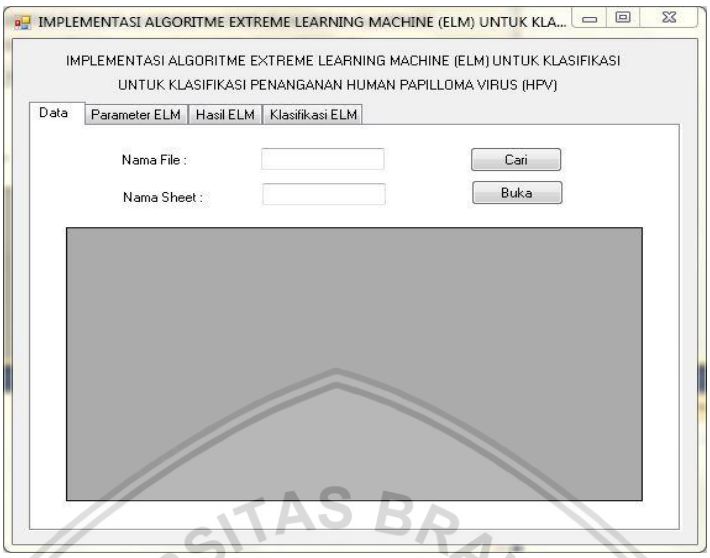
4.2 Perancangan Antarmuka Sistem

Tahapan perancangan antarmuka sistem dilakukan untuk merancang tampilan dari sistem yang akan diimplementasikan. antarmuka sistem yang akan dirancang terdiri dari 4 halaman. Halaman pertama merupakan halaman data yang ditujukan untuk proses memasukkan dataset yang ada untuk diproses menggunakan algoritme *Extreme Learning Machine.* Halaman kedua merupakan halaman yang ditujukan untuk memasukkan parameter yang nantinya digunakan pada proses latih dan proses uji menggunakan algoritme *Extreme Learning Machine*. Halaman ketiga merupakan halaman yang ditujukan untuk menampilkan hasil dari fase latih dan fase uji. Sedangkan halaman keempat merupakan halaman yang ditujukan untuk memasukkan parameter yang digunakan untuk klasifikasi dan hasil dari klasifikasi. Parameter masukan untuk klasifikasi antara lain meliputi *sex, age, time, number of warts, type, area*.

4.2.1 Rancangan Halaman Data

Rancangan halaman data merupakan halaman yang ditujukan untuk memasukkan dataset yang berupa *file Microsoft Excel* (.xls / .xlsx). Selain itu

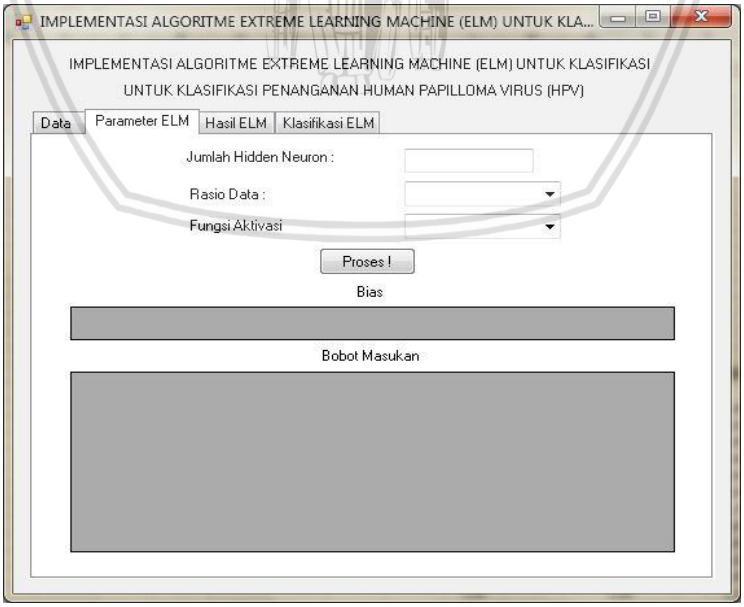
terdapat *form* untuk mengisi nama *sheet* yang terdapat di *file Excel* tersebut. Pada halaman ini menampilkan tabel dataset yang telah dimuat untuk proses *Extreme Learning Machine*. Rancangan halaman data ditampilkan pada Gambar 4.1.



Gambar 4.1 Rancangan Antarmuka Halaman Data

4.2.2 Rancangan Antarmuka Halaman Parameter ELM

Rancangan halaman parameter merupakan halaman untuk memasukkan parameter ELM yaitu fungsi aktivasi, jumlah *hidden neuron* dan rasio data latih dan uji. Pada halaman ini akan menampilkan matriks bobot masukan *W* dan matriks bias setelah tombol Proses ELM ditekan. Rancangan halaman parameter *Extreme Learning Machine* dapat dilihat pada Gambar 4.2.



Gambar 4.2 Rancangan Antarmuka Halaman Parameter ELM

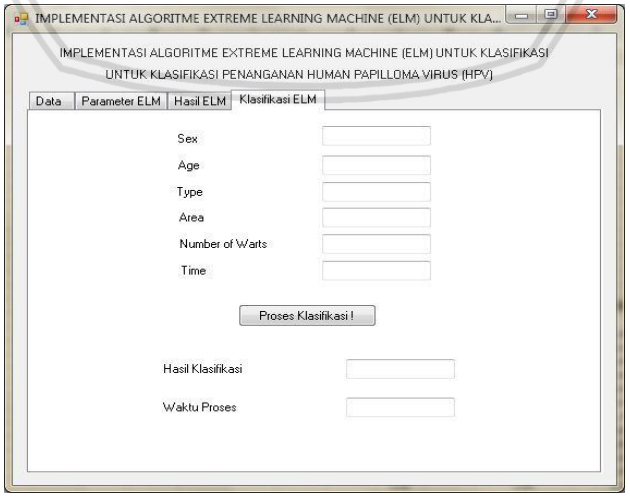
4.2.3 Rancangan Antarmuka Halaman Hasil ELM

Rancangan halaman hasil *Extreme Learning Machine* adalah halaman yang menampilkan hasil fase latih serta hasil fase uji. Pada hasil fase latih akan ditampilkan matriks bobot keluaran $\hat{\beta}$. Sedangkan pada hasil fase uji akan ditampilkan hasil klasifikasi *Result of Treatment* dari data uji. Rancangan halaman hasil *Extreme Learning Machine* dapat dilihat pada Gambar 4.3.

Gambar 4.3 Rancangan Antarmuka Halaman ELM

4.2.4 Rancangan Antarmuka Halaman Klasifikasi

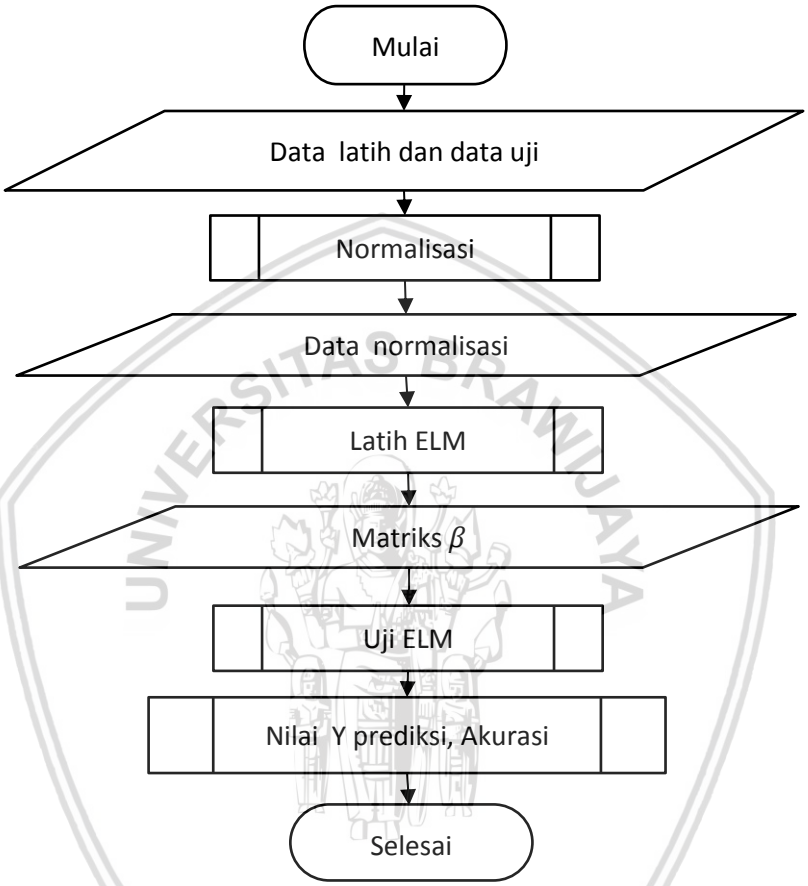
Rancangan halaman klasifikasi adalah halaman untuk meelakukan klasifikasi metode penangana HPV berdasarkan *Sex, Age, Time,Number of Warts, Type* dan, *Area*. Halaman ini menampilkan *form* untuk memasukan nilai *Sex, Age, Time,Number of Warts, Type* dan, *Area* serta hasil klasifikasi *Treatment*. Rancangan Halaman klasifikasi dapat dilihat pada Gambar 4.4.



Gambar 4.4 Rancangan Antarmuka Halaman Klasifikasi

4.3 Rancangan Algoritme *Extreme Learning Machine*

Pada sub bab 4.3 ini akan dijelaskan mengenai perancangan algoritme *Extreme Learning Machine* dalam bentk diagram alir untuk menjelaskan alur kerja sistem dalam mengklasifikasikan penanganan *Human Papilloma Virus* menggunakan algoritme *Extreme Learning Machine.* Diagram alir mengenai proses *Extreme Learning Machine* pada penelitian ini dapat dilihat pada Gambar 4.5.



Gambar 4.5 Diagram Alir Algoritme *Extreme Learning Machine*

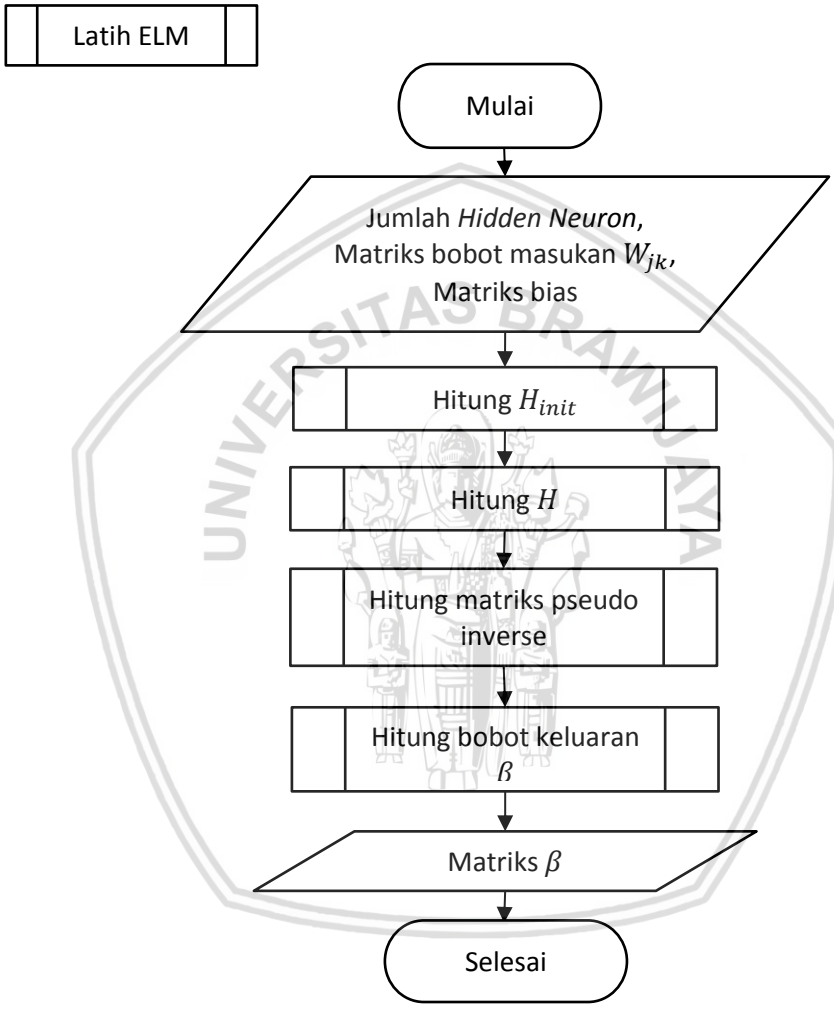
Proses klasifikasi yang akan dilakukan menggunakan algoritme *Extreme Learning Machine* pada penelitian ini adalah sebagai berikut:

1. Sistem yang dibangun akan menerima masukan berupa data latih dan data uji.
2. Data yang telah diterima akan dinormalisasi menggunakan normalisasi min maks dengan rentang antara 0 sampai dengan 1.
3. Data yang telah dinormalisasi akan masuk kedalam proses latih ELM.
4. Setelah proses latih ELM dilakukan akan menghasilkan keluaran yakni nilai $\hat{\beta}$. Nilai $\hat{\beta}$ nantinya akan digunakan pada proses uji ELM.
5. Setelah nilai $\hat{\beta}$ ditemukan, nilai $\hat{\beta}$ akan dugunakan pada proses uji ELM.

6. Setelah melakukan proses uji ELM, hasil $\hat{Y}$ akan diproses lagi untuk menghitung nilai akurasi dari proses uji yang dilakukan. Alur proses ELM selesai.

4.3.1 Proses Latih *Extreme Learning Machine*

Proses latih ELM akan ditunjukkan pada Gambar 4.6. pada proses latih, keluaran yang dihasilkan berupa nilai $\hat{\beta}$ yang nantinya digunakan pada proses uji. Berikut diagram alir proses latih pada algoritme *Extreme Learning Machine.*



Gambar 4.6 Diagram Alir Proses Latih ELM

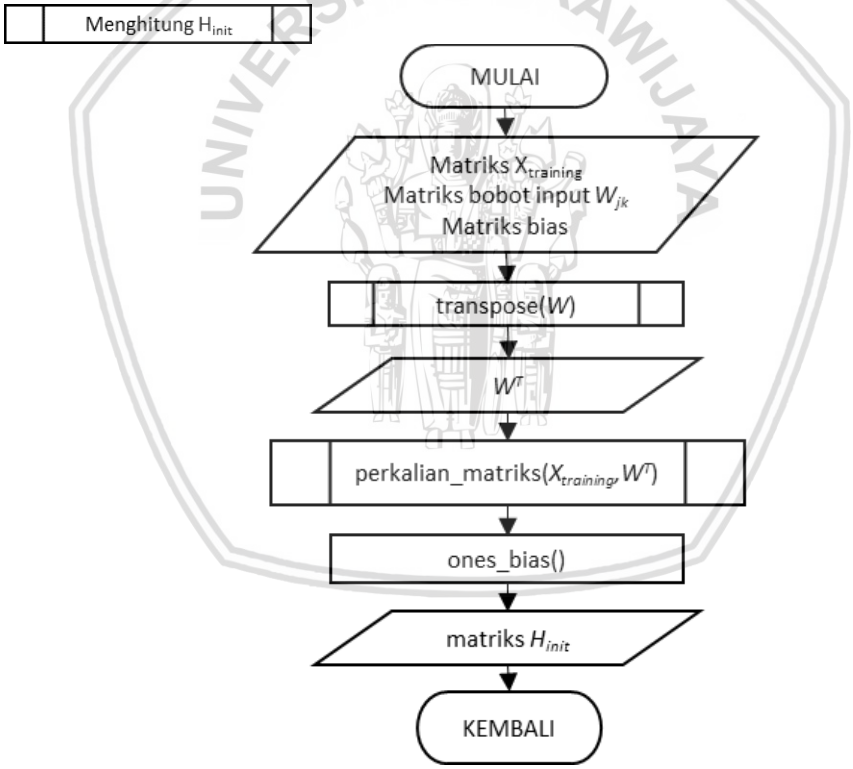
Langkah yang dilakukan pada proses latih seperti Gambar 4.6 adalah sebagai berikut:

1. Pada proses latih ELM, sistem akan menerima masukan jumlah *hidden neuron*, jumlah *hidden neuron* merefleksikan jumlah bias dan bobot masukan matriks W_{jk}.

2. Setelah mendapatkan bobot masukan dan bias yang memiliki rentang nilai antara -1 sampai 1 dan 0 sampai 1, selanjutnya kana dilakukan proses perhitungan H_{init}.
3. Setelah mendapatkan nilai H_{init} selanjutnya nilai tersebut akan digunakan untuk menghitung nilai H dimana nilai H_{init} kana diproses dengan fungsi aktivasi.
4. Setelah melakukan perhitungan nilai h selanjutnya dilakukan proses perhitungan matriks *Moore-Penrose pseudo Inverse.*
5. Setelah nilai matriks *Moore-Penrose Pseudo Inverse* didapatkan kemudian dilakukan proses perhitungan nilai $\hat{\beta}$.

4.3.1.1 Perhitungan H_{init}

Pada perhitungan H_{init} dilakukan perkalian matriks X_{latih} dengan matriks bobot W^T kemudian hasilnya akan ditambahkan dengan nilai bias. Berikut ini merupakan diagram alir pehitungan H_{init} yang ditunjukkan pada Gambar 4.7.

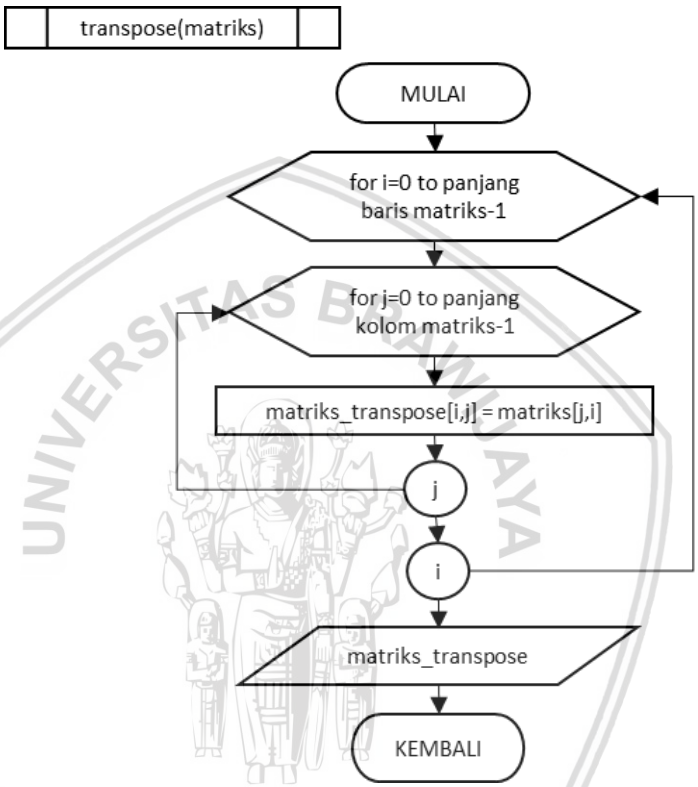


Gambar 4.7 Diagram alir Perhitungan Nilai H_{init}

Langkah yang dilakukan pada proses perhitungan H_{init} seperti Gambar 4.7 adalah sebagai berikut:

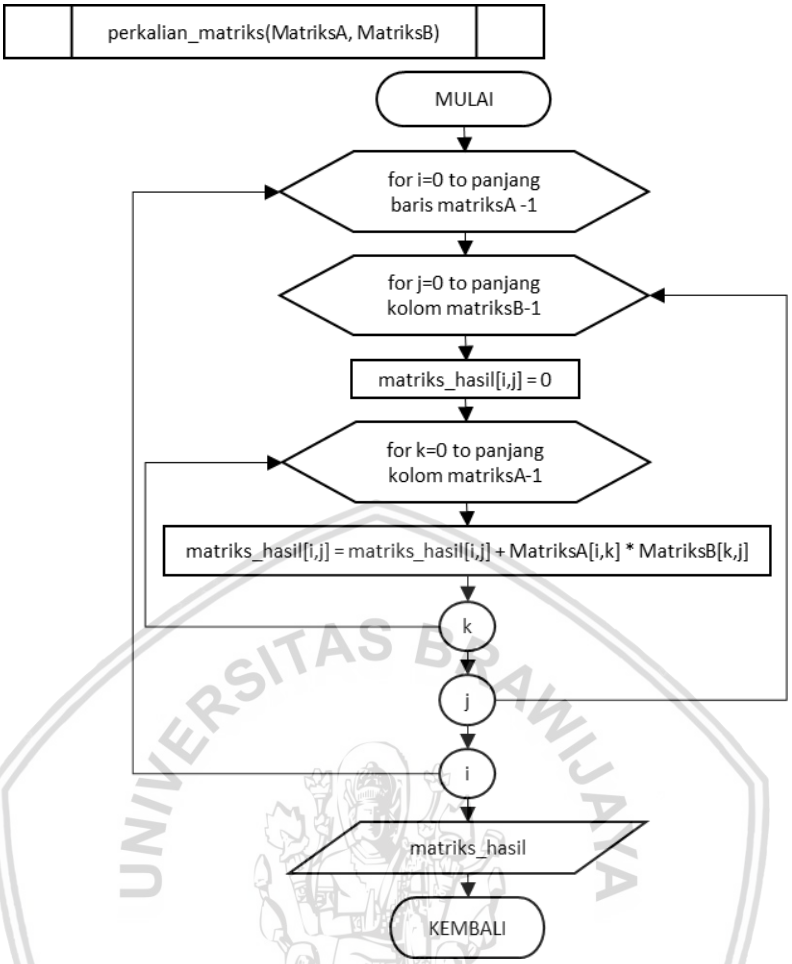
1. Pada proses ini dibutuhkan nilai $X_{latih,}$ bobot masukan dan bias untuk menghitung nilai $H_{init..}$

2. Setelah mendapatkan nilai yang dibutuhkan, kemudian dilakukan *transpose* nilai bobot masukan yang nantinya menghasilkan nilai *W* transpose.
3. Nilai *W* transpose yang didapat kemudian dengan nilai bobot masukan awal (*W*) yang setelah itu nilai perkalian matrikas yang didapat kana titambahkan dengan nulai bias yang sudah dibuat secara acak. Nilai penjumlahan bobot *W* dengan bias yang didapat kana digunakan pada proses perhitungan nilai *H.*



Gambar 4.8 Diagram Alir Proses Transpose Matriks

1. Sistem menerima masukan yang akan ditranspose
2. Proses transpose dilakuakan dengan merrubah baris dan kolom masukan menjasi kolom dan baris baru yang nantinya disimpan pada matriks_transpose.
3. Setelah itu fungsi ini akan mengembalikan nilai matriks transpose.



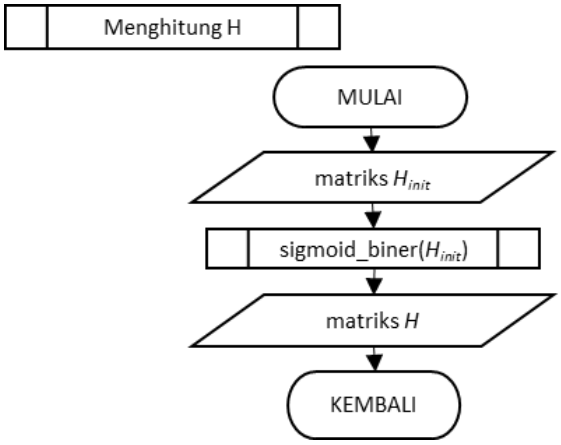
Gambar 4.9 Diagram Alir Perkalian Matriks

Langkah yang dilakukan pada proses perkalian matriks seperti Gambar 4.9 adalah sebagai berikut:

1. Sistem menerima masukan berupa 2 matriks yang akan dikalikan.
2. Melakukan perkalian matriks dengan mengalikan baris matriksA dengan kolom matriksB sebanyak jumlah matriksA yang digunakan.
3. Fungsi perkalian ini akan mengembalikan nilai matriks hasil.

4.3.1.2 Perhitungan Nilai H

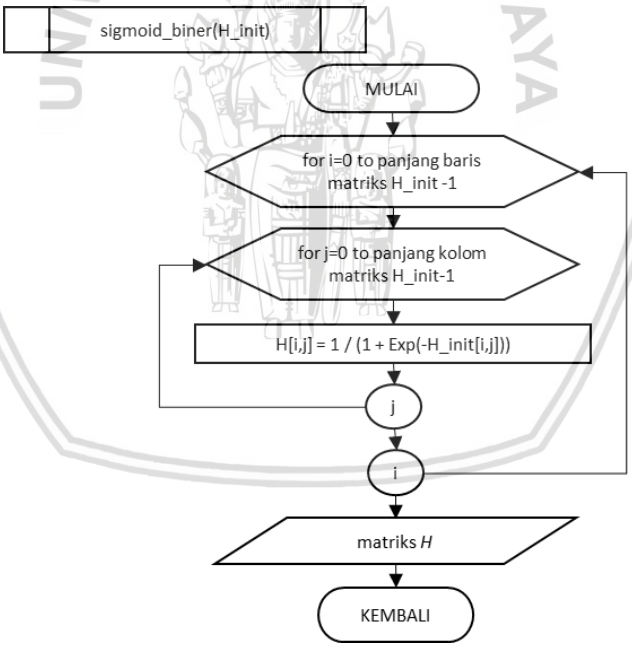
Pada perhitungan nilai *H* dilakukan dengan melakukan perhitungan keluaran H dengan menggunakan fungsi aktivasi, fungsi aktivasi yang digunakan adalah sigmoid biner. Perhitungan nilai *H* akan ditampilkan pada Gambar 4.10 sebagai berikut.



Gambar 4.10 Diagram Alir Perhitungan *H*

Langkah yang dilakukan pada proses perhitungan *H* seperti Gambar 4.10 adalah sebagai berikut:

1. Sistem mendapat masukan berupa matriks H_{init} kemudian nilai tersebut akan diproses dengan menggunakan fungsi aktivasi *Sigmoid Biner*
2. Setelah dilakukan perhitungan kemudian fungsi akan mengembalikan nilai *H*



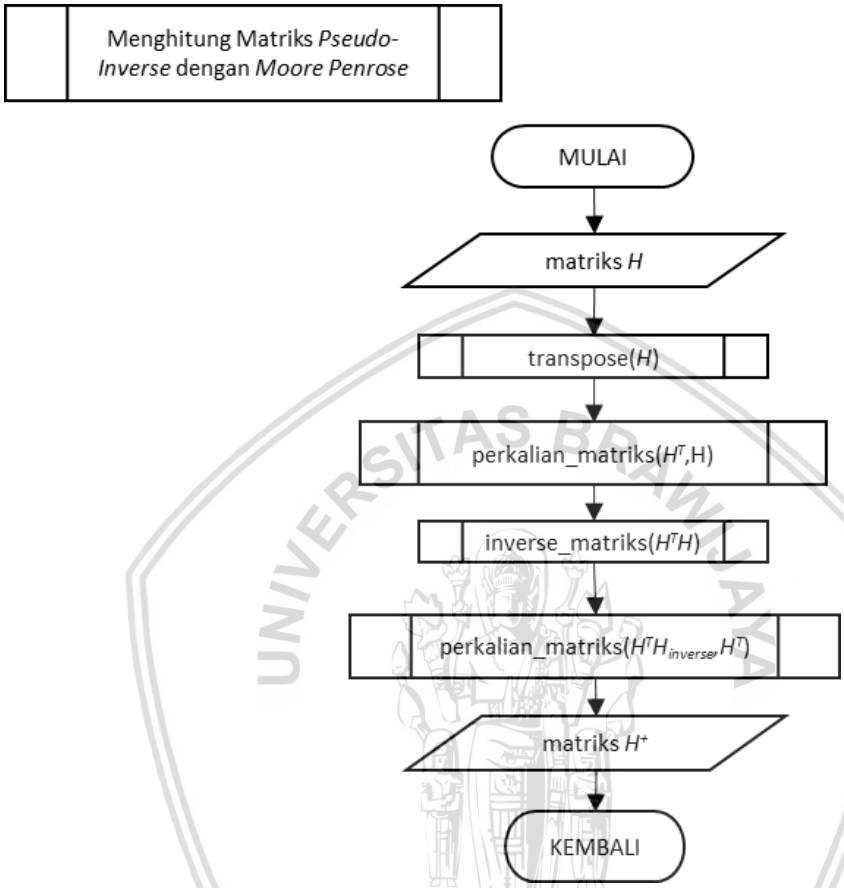
Gambar 4.11 Diagram Alir Fungsi Sigmoid Biner

Langkah yang dilakukan pada proses perhitungan Fungsi Aktivasi *Sigmoid Biner* seperti Gambar 4.11 adalah sebagai berikut:

1. Sistem mendapatkan masukan nilai H_{init} kemudian dilakukan perhitungan dengan menggunakan fungsi aktivasi *sigmoid Biner.*
2. Setelah dilakukan perhitungan fungsi aktivasi *Sigmoid Biner* kemudian akan mengembalikan nilai matriks *H.*

4.3.1.3 Perhitungan Matriks $\mathbf{H}^{+}$ dengan Moore-Penrose

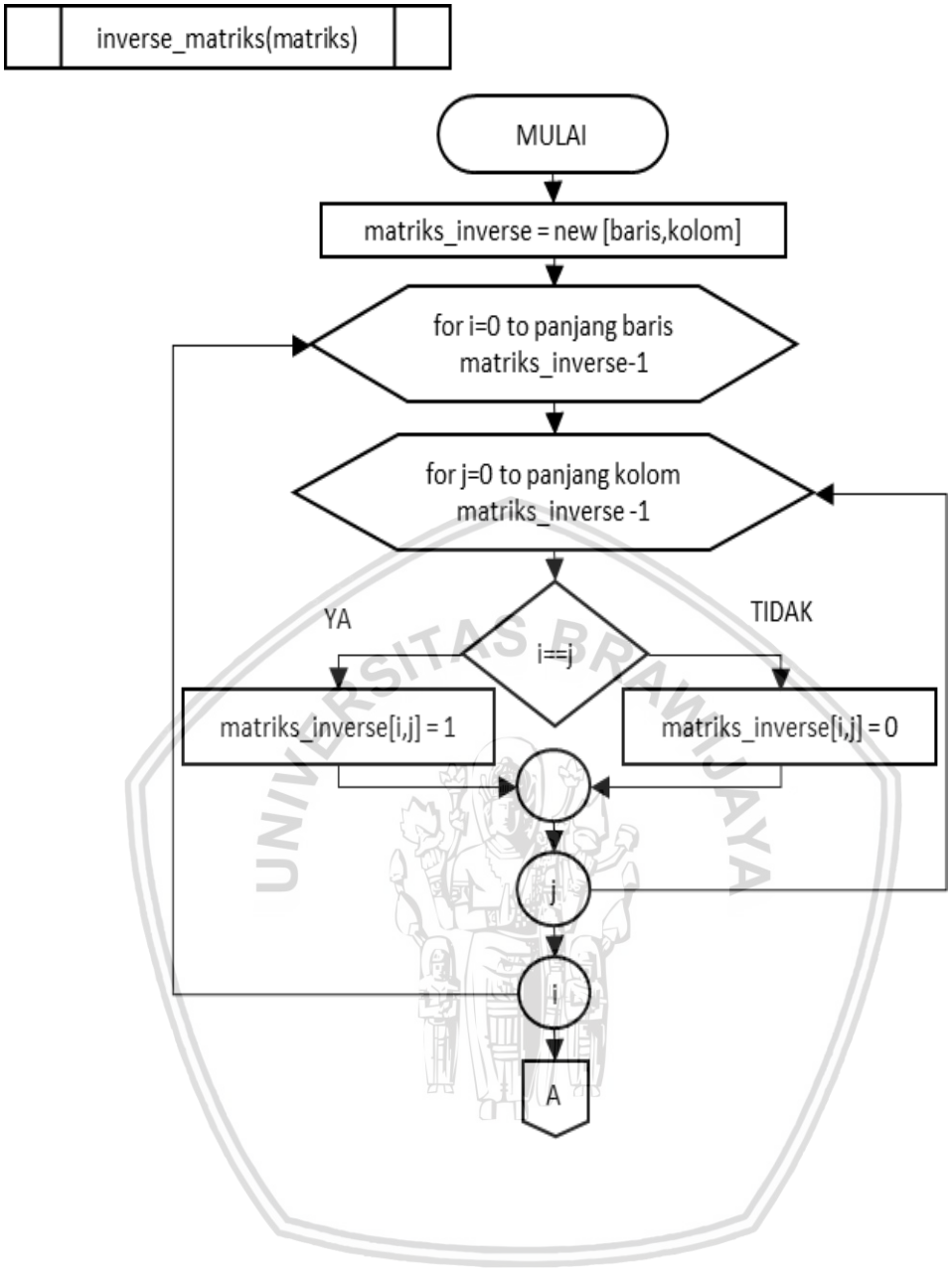
Diagram alir pada Gambar 4.12 akan ditunjukkan perhitungan nilai matriks H^{+}. Berikut ini merupakan diagram alir dari proses perhitungan tersebut:

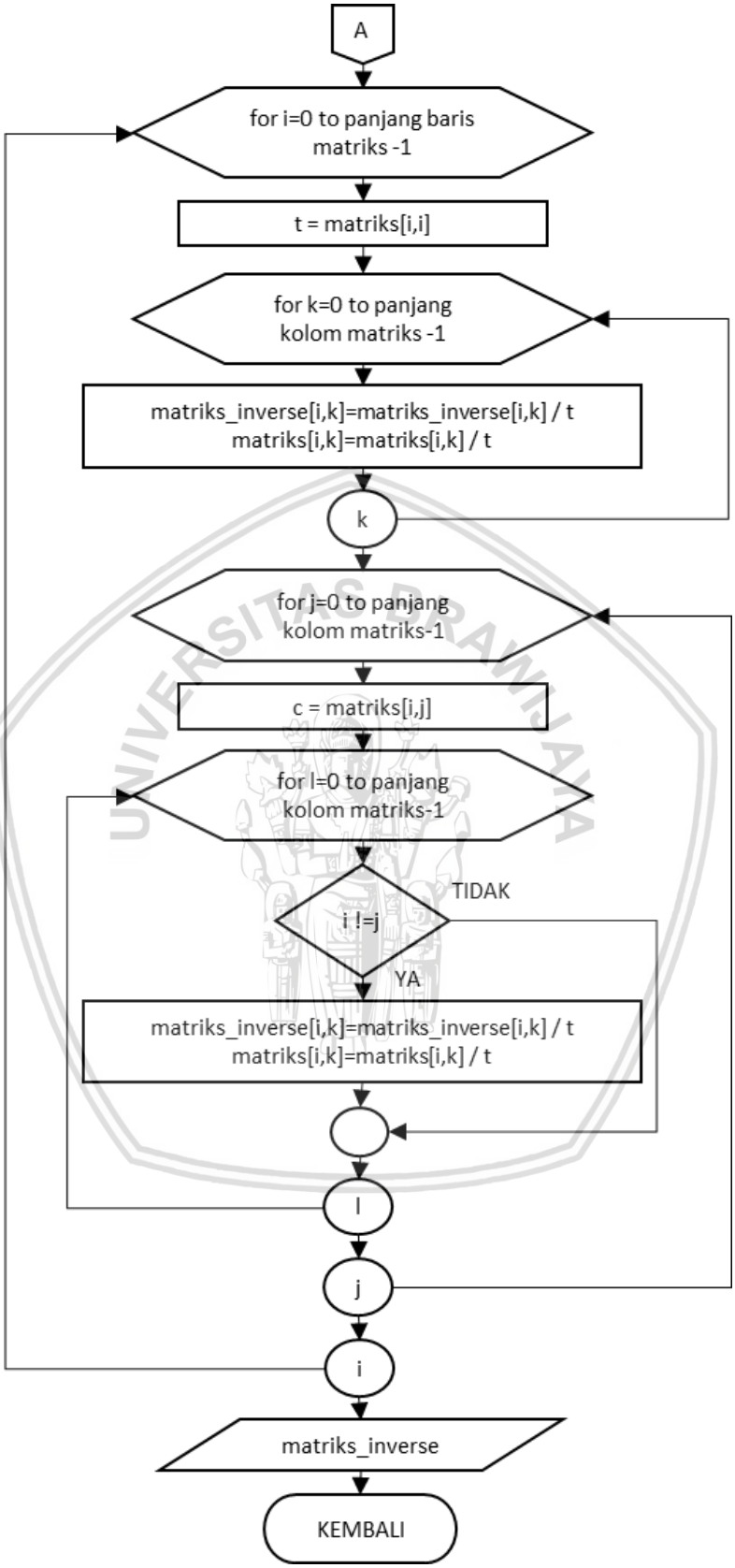


Gambar 4.12 Diagram Alir Perhitungan Matriks *H⁺Moore-Penrose*

Langkah yang dilakukan pada proses perhitungan matriks H^{+} *Moore Penrose* seperti Gambar 4.12 adalah sebagai berikut:

1. Pada perhitungan ini, diperlukan nilai matriks *H* kemudian nilai dai matriks *H* akan ditranspose.
2. Matriks *H* yang telah ditranspose akan dikalikan dengan matriks *H* yang menghasilkan matriks H^TH.
3. Matriks H^TH akan dilakukan proses inverse yang menghasilkan nilai matriks $H^TH_{inverse.}$
4. Setelah itu dilakukanperkalian terhadap nilai matriks $H^TH_{inverse}$ dengan matriks H^T. Kemudian nilai hasil perkalianakan dikembalikan dengan nilai matriks H^{+}.


inverse_matriks(matriks)
MULAI
matriks_inverse = new [baris,kolom]
for i=0 to panjang baris matriks_inverse-1
for j=0 to panjang kolom matriks_inverse -1
i==j
YA
TIDAK
matriks_inverse[i,j] = 1
matriks_inverse[i,j] = 0
j
i
A



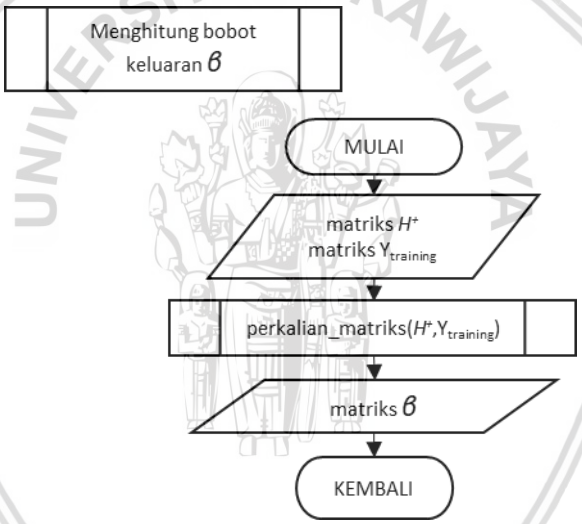
Gambar 4.13 Diagram Alir Inverse Matriks

Langkah yang dilakukan pada proses perhitungan inverse matriks seperti Gambar 4.13 adalah sebagai berikut:

1. Mendapat masukan berupa metriks yang akan diproses inverse.
2. Membuat matriks baru berupa matriks identitas dengan panyang baris dan kolom serupa dengan matriks masukan.
3. Melaukan proses inverse dengan operasi baris elementer.
4. Fungsi inverse akan mengembalikan nilai matriks inverse yang sudah di proses.

4.3.1.4 Perhiutngan Matriks Bobot Keluaran ($\hat{\beta}$)

Diagram alir pada Gambar 4.14 akan ditunjukkan perhitungan nilai matriks H^+. Pada proses perhitungan matriks bobt keluaran, nilai matriks H^+ akan dilakukan perkalian dengan nilai Y_{latih}. Hasil dari proses perkalian yang dilakukan akan mengahasilakan nilai $\hat{\beta}$.Berikut ini merupakan diagram alir dari proses perhitungan tersebut:



Gambar 4.14 Diagram Alir Perhitungan Matriks Bobot Keluaran $\hat{\beta}$

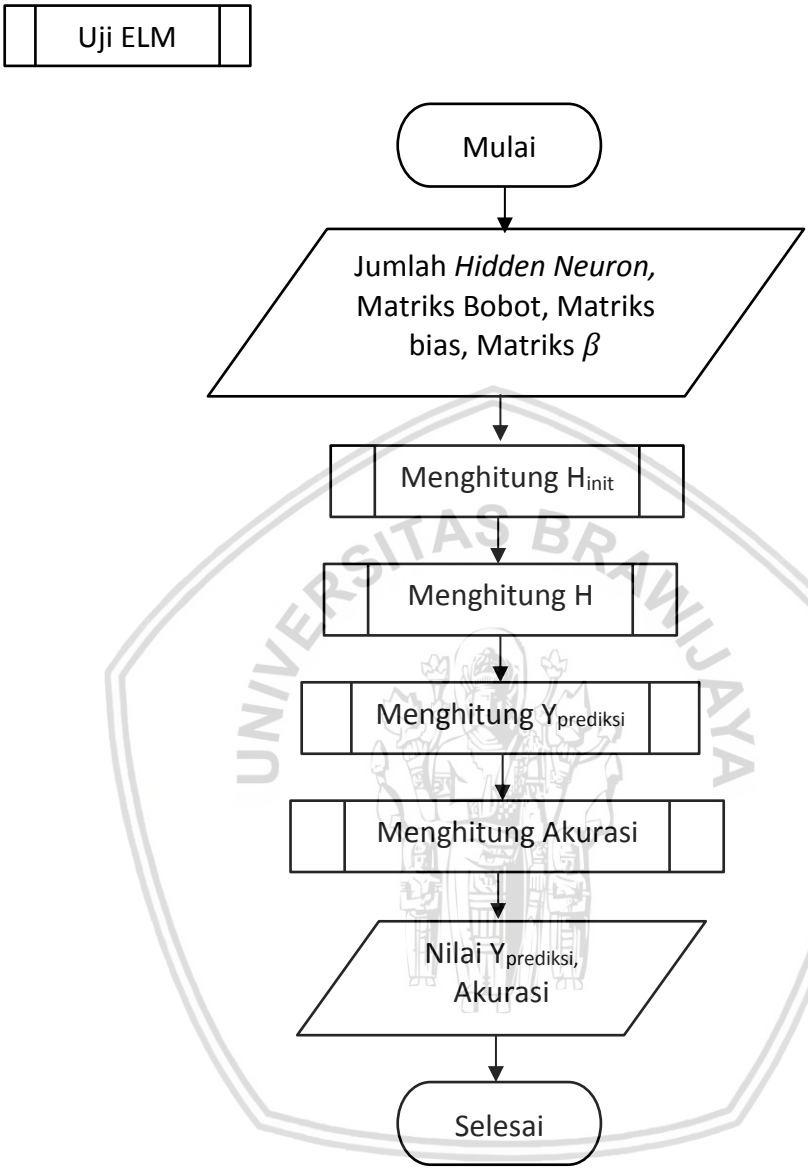
Langkah yang dilakukan pada proses perhitungan matriks bobot keluaran $\hat{\beta}$ seperti Gambar 4.13 adalah sebagai berikut:

1. Mendapat masukan berupa nilai H^+*Moore-Penrose Pseudo Invers* dan target *latih* Y_{latih}.
2. Nilai H^+*Moore-Penrose Pseudo Invers* dikalikan dengan target Y_{latih} dengan nilai keluaran berupa $\hat{\beta}$.
3. Proses ini akan mengembalikan nilai berupa nilai matriks $\hat{\beta}$.

4.3.2 Proses Uji *Extreme Learning Machine*

Tahapan yang dilakukan setelah proses latih ELM yang menghasilkan nilai bobot keluaran $\hat{\beta}$ adalah proses uji. Pada proses uji diperlukan bobot masukan dan

bias yang sama seperti pada proses latih ELM. Berikut ini merupakan diagram alir dari proses perhitungan tersebut yang tertera pada Gambar 4.15.



Gambar 4.15 Diagram Alir Proses Uji ELM

Langkah yang dilakukan pada proses Uji *Extreme Learning Machine* seperti Gambar 4.15 adalah sebagai berikut:

1. Mendapat masukan berupa nilai $X_{uji,}$ matriks masukan $W_{jk,}$ matriks bias dan nilai β.
2. Melakukan proses hitung H_{init} dengan nilai yang digunakan nilai $X_{uji.}$
3. Melakukan perhitungan matriks H kemudian menghirung matriks $Y_{prediksi}$ dengan menggunakan nilai β yang dikalikan dengan nilai matriks H.
4. Nilai $\hat{Y}$ didapatkan dari hasil perkalian matriks H dengan $\hat{\beta}$.

5. Setelah didapatkan nilai $\hat{Y}$ kemudian akan dilakukan perhitungan akurasi.
6. Proses ini akan mengembalikan nilai berupa nilai $\hat{Y}$ dan nilai akurasi.

4.4 Perhitungan Manual

4.4.1 Normalisasi Data

Normalisasi data yang digunakan pada penelitian ini adalah menggunakan normalisasi min maks. Pada normalisasi yang dilakuakan, data akan dinormalisasi pada rentang nilai anatara 0 ,01sampai 0,99. Berikut merupakan contoh perhitungan normalisasi yang dilakukan.

$$y = \frac{x - min}{maks - min} * (B_{atas} - B_{bawah}) + B_{bawah}$$

$$y = \frac{19 - 15}{67 - 15} * (0.99 - 0.01) + 0.01$$

$$y = \frac{4}{52} * (0.99 - 0.01) + 0.01$$

$$y = 0.076 * 0.98 + 0.01$$

$$y = 0.075 + 0.01$$

$$y = 0.08448$$

Berikut pada Tabel 4.2 da Tabel 4.3 ditunjukkan data latih dan data uji yang telah dinormalisasi.

Tabel 4.2 Data Latih Normalisasi

Sex	Age	Time	Number of Warts	Type	Area	Result of Treatment
0.01	0.09	0.66	0.39	0.01	0.18	1.00
0.01	0.03	0.91	0.12	0.50	0.07	1.00
0.01	0.18	0.78	0.39	0.01	0.03	1.00
0.01	0.39	0.76	0.45	0.01	0.12	1.00
0.99	0.14	0.41	0.45	0.01	0.08	1.00
0.99	0.01	0.11	0.61	0.99	0.08	1.00
0.99	0.05	0.43	0.12	0.01	0.07	1.00
0.99	0.01	0.49	0.06	0.01	0.04	1.00
0.01	0.27	0.72	0.12	0.01	0.56	2.00
0.01	0.24	0.41	0.06	0.01	0.03	2.00
0.01	0.03	0.86	0.06	0.01	0.12	2.00
0.01	0.27	0.41	0.61	0.99	0.09	2.00
0.99	0.69	0.32	0.01	0.01	0.08	2.00

0.99	0.41	0.14	0.50	0.99	0.05	2.00
0.99	0.73	0.59	0.28	0.01	0.09	2.00
0.99	0.09	0.49	0.06	0.01	0.25	2.00
0.01	0.91	0.22	0.12	0.99	0.03	3.00
0.01	0.99	0.30	0.55	0.99	0.03	3.00
0.01	0.16	0.84	0.34	0.99	0.08	3.00
0.01	0.41	0.91	0.06	0.01	0.01	3.00
0.99	0.33	0.99	0.61	0.99	0.83	3.00
0.99	0.33	0.99	0.17	0.99	0.83	3.00
0.99	0.37	0.99	0.12	0.99	0.11	3.00
0.99	0.16	0.78	0.23	0.99	0.08	3.00

Tabel 4.3 Data Uji Normalisasi

Sex	Age	Time	Number of Warts	Type	Area	Result of Treatment
0.01	0.10	0.36	0.61	0.01	0.01	1.00
0.01	0.26	0.39	0.12	0.01	0.12	1.00
0.99	0.01	0.16	0.50	0.99	0.08	1.00
0.99	0.01	0.34	0.01	0.01	0.01	1.00
0.01	0.20	0.47	0.06	0.01	0.33	2.00
0.01	0.12	0.66	0.12	0.01	0.02	2.00
0.99	0.61	0.43	0.12	0.99	0.03	2.00
0.99	0.03	0.82	0.34	0.01	0.16	2.00
0.01	0.37	0.99	0.01	0.99	0.17	3.00
0.01	0.16	0.84	0.34	0.99	0.08	3.00
0.99	0.48	0.72	0.28	0.50	0.09	3.00
0.99	0.39	0.68	0.39	0.99	0.12	3.00

4.4.2 Proses Latih

Pada tahapan latih atau proses pembelajaran adalah proses dimana proses yang dilakukan untuk memperoleh bobot yang sesuai yang nantinya akan digunakan pada proses uji. pada tahapan ini data yang akan digunakan sebanyak 24 data dengan *hidden neuron* sebanyak 3 dan menggunakan fungsi aktivasi *sigmoid biner*. Parameter yang digunakan pada tahapan *latih* abtara lain berupa matriks bobot masukan W, dan matriks bias. Matriks bobot masukan W memiliki ordo jumlah *hidden neuron* $\times$ *input layer*, dengan jumlah *input layer* sebanyak 6.

6 *input layer* yang digunakan pada proses *latih* adalah *Sex, Age, Time, Number of Warts, Type, Area, Result of Treatment*.

Selain matriks bobot masukan W juga terdapa tmatriks bias. Matriks bias memiliki ordo 1 ×*hidden neuron* yang mana pada proses latih menggunakan *hidden neuron* sebanyak 3. sebelum melakukan proses latih, data latih yang telah dinormalisasi dibagi menjadi X_{latih} dan $Y_{latih.}$ X_{latih} digunakan sebagai *input layer* danY_{latih}sebagai target. Data yang digunakan sebagai X_{latih} dapat dilihat pada Tabel 4.4.

Tabel 4.4 Data Xl_{atih}

X_{latih}	Sex	Age	Time	Number of Warts	Type	Area
1	0,99	0,01	0,16	0,12	0,01	0,01
2	0,99	0,29	0,56	0,06	0,01	0,01
3	0,01	0,19	0,10	0,12	0,99	0,06
4	0,99	0,06	0,44	0,12	0,01	0,07
5	0,01	0,39	0,07	0,06	0,01	0,10
6	0,99	0,04	0,71	0,01	0,50	0,07
7	0,99	0,91	0,50	0,28	0,01	0,09
8	0,99	0,01	0,67	0,61	0,01	0,04
9	0,01	0,91	0,73	0,06	0,50	0,07
10	0,01	0,39	0,01	0,50	0,01	0,13
11	0,01	0,59	0,63	0,39	0,50	0,07
12	0,01	0,04	0,88	0,06	0,01	0,12
13	0,01	0,01	0,88	0,55	0,01	0,04
14	0,01	0,14	0,31	0,55	0,01	0,09
15	0,99	0,24	0,35	0,01	0,01	0,20
16	0,01	0,01	0,16	0,55	0,01	0,01
17	0,99	0,34	0,63	0,17	0,01	0,02
18	0,99	0,14	0,29	0,28	0,01	0,09
19	0,99	0,01	0,46	0,61	0,01	0,06
20	0,01	0,09	0,33	0,01	0,01	0,09
21	0,01	0,01	0,24	0,06	0,99	0,99
22	0,01	0,06	0,48	0,50	0,01	0,07
23	0,99	0,34	0,41	0,45	0,01	0,12
24	0,01	0,01	0,33	0,34	0,01	0,01

Data Y_{latih} digunakan sebagai target dapat dilihat pada tabel 4.5 sebagai berikut:

Tabel 4.5 Data Y_{latih}

Y_{latih}	Result of Treatment
1	1
2	2
3	2
4	1
5	2
6	1
7	2
8	1
9	2
10	1
11	2
12	2
13	1
14	1
15	2
16	1
17	2
18	1
19	2
20	1
21	2
22	1
23	1
24	1

4.4.2.1 Inisialisasi Bobot Masukan dan Bias

Proses inisialisasi untuk bobot masukan dan bias nilainya akan didapat secara acak. Pada inisialisasi bobot masukan nilai akan dibuat secara acak dengan rentang nilai -1 sampai dengan 1 dan nilai bias dengan rentang 0 sampai dengan 1. Pada penelitian ini yang disampaikan dalam laporan menggunakan 3 *hidden neuron*.

Pada marrtiks bobot, bobot masukan merupakan matriks dengan ordo 3 x 6 dan matriks bias dengan ordo 1 x 3. Matriks bobot dam matriks bias dapat dilihat pada Tabel 4.6 dan Tabel 4.7.

Tabel 4.6 Tabel Bobot Masukan

W	Sex	Age	Time	Number of Warts	Type	Area
1	0.54	0.58	0.87	0.42	0.56	0.61
2	0.39	0.69	0.69	0.10	0.45	0.14
3	0.63	0.98	0.50	0.79	0.17	0.19

Tabel 4.7 Tabel Bobot Bias

Bias	1	2	3
1	0.16	0.20	0.53

4.4.2.2 Proses Perhitungan Matriks H_{init}

Pada perhitungan nilai matriks H_{init} memerlukan niali masukan berupa transpose bobot masukan. Nilai transpose bobot masukan akan dilakukan perkalian dengan $X_{latih.}$ Nilai matriks bobt yang telah ditranspose dan matriks H_{init} dapat ilihat pada Tabel 4.8 dan 4.9.

Tabel 4.8 Tabel W transpose

W	1	2	3
Sex	0.54	0.39	0.63
Age	0.58	0.69	0.98
Time	0.87	0.69	0.50
Number of Warts	0.42	0.10	0.79
Type	0.56	0.45	0.17
Area	0.61	0.14	0.19

$$H_{init(1,1)} = X_{training}.W^T + b$$

=((0.01*0.54)+(0,09*0.58)+ (0.66*0.87) + (0.39*0.42) + (0.01*0.56) + (0.18*0.61)) + 0.16

=(0.0058 + 0.0522 + 0.574 + 0.163 + 0.0056 + 0.11) + 0.16

= 0.910 + 0.16

= 1.07

Tabel 4.9 Matriks H_{init}

H_{init}	1	2	3
1	1.07	0.78	1.29
2	1.35	1.09	1.20
3	1.14	0.91	1.41
4	1.32	1.05	1.66
5	1.38	1.02	1.86
6	1.67	1.18	1.87
7	1.20	0.94	1.52
8	1.19	0.94	1.46
9	1.35	0.98	1.36
10	0.71	0.66	1.02
11	1.04	0.84	1.07
12	1.55	1.19	1.66
13	1.44	1.30	2.00
14	1.86	1.46	2.18
15	1.82	1.54	2.39
16	1.36	1.02	1.57
17	1.52	1.44	1.80
18	1.82	1.60	2.26
19	1.75	1.38	1.56
20	1.24	1.12	1.44
21	3.07	2.11	2.76
22	2.89	2.07	2.42
23	2.45	1.99	2.28
24	2.18	1.71	2.05

4.4.2.3 Proses Hitung Matriks *H*

Proses yang dilakukan setelah menemukan nilai H_{init} adalah melakukan proses perhitungan nilai *H* dengan fungsi aktivasi. Contoh perhitungan nilai *H* akan ditampilkan pada perhitungan dibawah dan hasil perhitungan *H* dapat dilihat pada Tabel 4.10.

$$H = \frac{1}{1 + \exp(-(X \,.\, W^T) + b)}$$

$$H = \frac{1}{1 + \exp(-H_{init})}$$

$$H_{(1,1)} = \frac{1}{1 + \exp(-1.07)}$$

$$= 0.74$$

Tabel 4.10 Matriks Keluaran *Hidden Neuron*

H	1	2	3
1	0.74	0.69	0.78
2	0.79	0.75	0.77
3	0.76	0.71	0.80
4	0.79	0.74	0.84
5	0.80	0.74	0.86
6	0.84	0.77	0.87
7	0.77	0.72	0.82
8	0.77	0.72	0.81
9	0.79	0.73	0.80
10	0.67	0.66	0.74
11	0.74	0.70	0.74
12	0.83	0.77	0.84
13	0.81	0.79	0.88
14	0.86	0.81	0.90
15	0.86	0.82	0.92
16	0.80	0.74	0.83
17	0.82	0.81	0.86
18	0.86	0.83	0.91
19	0.85	0.80	0.83
20	0.77	0.75	0.81
21	0.96	0.89	0.94
22	0.95	0.89	0.92
23	0.92	0.88	0.91
24	0.90	0.85	0.89

4.4.2.4 Proses Perhitungan Matriks H^+ *Pseudo Inverse* dengan *Moore-Penrose*

Proses yang dilakukan setelah proses perhitungan *H* adalah menghitung menghitung matriks H^+ *Pseudo-Inverse* dengan *Moore-Penrose*. Hasil perhitungan $H(H^T)$ ditampilkan pada Tabel 4.11.

$$H^+ = (H^T \,.\, H)^{-1}.H^T$$

Tabel 4.11 Matriks *Transpose* Keluaran *Hidden Neuron* H

H^T	1	2	3	...	22	23	24
1	0.74	0.79	0.76	...	0.95	0.92	0.90
2	0.69	0.75	0.71	...	0.89	0.88	0.85
3	0.78	0.77	0.80	...	0.92	0.91	0.89

Proses yang dilakukan pada setelah mendapatkan matriks H^T adalah melakuak perkalian H^T dengan nilai *H* yang menghasilkan matriks dengan ordo 3x3. Hasil

perhitungan ditampilkan pada Tabel 4.12 dan contoh perhitungan dapat dilihat dibawah.

$= H^T . H$

= (0.74*0.74) + (0.79*0.69) + (0.76*0.78) + … + (0.95*0.90) + (0.92*0.85) + (0.90*0.89)

= 16.19

Tabel 4.12 Matriks Hasil Perkalian H^Tdengan *H*

$H^T.H$	1	2	3
1	16.19	15.27	16.65
2	15.27	14.40	15.71
3	16.65	15.71	17.16

Setelah mendapatkan nilai hasil perkalian $H(H^T)$ kemudian dilakukan proses inverse matriks menggunakan inverse Operasi Baris Elementer (OBE). Perhitungan inverse matriks menggunakan inverse OBE dan Tabel 4.13 yang berisi hasil inverse dapat dilihat dibawah.

1. Bentuk matriks identitas (*I*)

$$[H \mid I] = \begin{bmatrix} 16.19 & 15.27 & 16.65 \\ 15.27 & 14.40 & 15.71 \\ 16.65 & 15.71 & 17.16 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2. R1 : 16.19→ R1

$$[H \mid I] = \begin{bmatrix} 1 & 0{,}944 & 1.031 \\ 15.27 & 14.40 & 15.71 \\ 16.65 & 15.71 & 17.16 \end{bmatrix} \begin{bmatrix} 0.0618 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3. R2 – (15.27 x R1) → R2

$$[H \mid I] = \begin{bmatrix} 1 & 0.944 & 1.0315 \\ 0 & 0.0109 & -0.0019 \\ 16.65 & 15.71 & 17.16 \end{bmatrix} \begin{bmatrix} 0.061 & 0 & 0 \\ -0.9445 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4. R3 – (16.65 x R1) → R3

$$[H \mid I] = \begin{bmatrix} 1 & 0.9449 & 1.0315 \\ 0 & 0.01098 & -0.0019 \\ 0 & -0.0019 & 0.0039 \end{bmatrix} \begin{bmatrix} 0.0618 & 0 & 0 \\ -0.9449 & 1 & 0 \\ -1.0315 & 0 & 1 \end{bmatrix}$$

5. R2 : 0.009 → R2

$$[H \mid I] = \begin{bmatrix} 1 & 0.9449 & 1.0315 \\ 0 & 1 & -0.1740 \\ 0 & -0.019 & 0.0039 \end{bmatrix} \begin{bmatrix} 0.0618 & 0 & 0 \\ -86.028 & 91.042 & 0 \\ -1.031 & 0 & 1 \end{bmatrix}$$

6. R1 – (1.128 x R2) → R1

$$[H \mid I] = \begin{bmatrix} 1 & 0 & 1.196 \\ 0 & -0.0019 & 0.00033 \\ 0 & -0.0019 & 0.0039 \end{bmatrix} \begin{bmatrix} 81.352 & -86.0281 & 0 \\ 0.164 & -0.174 & 0 \\ -1.031 & 0 & 1 \end{bmatrix}$$

7. R3 – (0.003 x R2) → R3

$$[H \mid I] = \begin{bmatrix} 1 & 0 & 1.196 \\ 0 & 1 & -0.174 \\ 0 & 0 & 0.0035 \end{bmatrix} \begin{bmatrix} 81.352 & -86.028 & 0 \\ -86.028 & 91.042 & 0 \\ -334.858 & 48.738 & 279.968 \end{bmatrix}$$

8. R3 : -3.685 → R3

$$[H \mid I] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -0.174 \\ 0 & 0 & -0.174 \end{bmatrix} \begin{bmatrix} 156.515 & -141.759 & -21.78 \\ -141.759 & -334.85 & 23.49 \\ -21.78 & 23.49 & 48.73 \end{bmatrix}$$

9. R1 – (0.828 x R3) → R1

$$[H \mid I] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 156.616 & -144.321 & -23.497 \\ -144.321 & 91.042 & 279.41 \\ -21.78 & -23.497 & 42.658 \end{bmatrix}$$

10. R2 – (0.400 x R3) → R2

$$[H \mid I] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 156.616 & -141.759 & -21.78 \\ -141.759 & 175.613 & -23.479 \\ -21.78 & -23.497 & 42.658 \end{bmatrix}$$

Tabel 4.13 Matriks *Inverse* Hasil Perkalian H^+ dengan *H*

H^+	1	2	4
1	156.616	-141.759	-21.78
2	-141.759	175.613	-23.497
3	-21.78	-23.497	42.658

Hasil inverse matriks yang didapatkan kemuduan digunakan untuk menghitung nilai H^+, contoh perhitungan dan hasil H^+ ditampilkan dibawah.

$H^{+}_{(1,1)}$ = (-18506.032*0.5893) + (-9103.050*0.6819) + (22482.50*0.7635)

= 53.608

Tabel 4.14 Matriks *pseudo-inverseH*$^{+}$

H+	1	2	3	…	13	14	15
1	53.608	156.510	-130.946	…	-45.583	-59.315	-66.743
2	28.426	74.357	-63.357	…	-29.940	-26.961	-36.413
3	-66.853	-187.733	158.241	…	62.092	70.193	84.330

4.4.2.5 Proses Hitung Nilai $\hat{\beta}$

Nilai H^{+} yang telah didapatkan kemudian digunakan pada perhitungan nilai keluaran $\hat{\beta}$ yang mana pada proses ini nilai keluaran $\hat{\beta}$ didapat dari hasil kali nilai H^{+} dengan matriks target $Y_{latih.}$ Hasil nilai keluaran $\hat{\beta}$ akan ditampilkan pada Tabel 4.15 dan contoh perhitungan akan ditampilkan dibawah.

$$\beta = H^{+}.Y_{training}$$

$\beta_{(1,1)}$ = (53.608*0.2246) + (156.51*0.2157) + (-130.946*0.1712) + … +(-45.583*0.4827) + (-59.315*0.4827) + (-66.743*0.3314)

$\beta_{(1,1)}$ = -16.55

Tabel 4.15 Matriks Keluran $\hat{\beta}$

$\hat{\beta}$	1
1	-14.12
2	31.96
3	-13.17

4.4.3 Proses Uji *Extreme Learning Machine*

Pada tahapan uji, data yang digunakan sebanyak 12 data dengan jumlah *hidden neuron* seperti pada proses latih yaitu 3 serta menggunakan fungsi aktivasi *sigmoid biner.* Parameter yang dibutuhkan untuk melakukan proses uji antara lain, matriks bobot masukan *W*, dan matriks bias. Matriks bobot masukan dan bias yang digunakan pada tahap uji ini sama dengan matriks bobot masukan dan bias yang digunakan pada tahap latih. Sebelum melakukan proses uji, data uji yang telah dinormalisasi dibagi menjadi X_{uji} dan Y_{uji}. X_{uji} digunakan sebagai *input layer* sedangkan Y_{uji} sebagai target. Data X_{uji} dan Y_{uji}dapat dilihat pada Tabel 4.16 dan Tabel 4.17.

Tabel 4.16 Data X_{uji}

X_{uji}	1	2	3	4	5	6
1	0.01	0.10	0.36	0.61	0.01	0.01
2	0.01	0.26	0.39	0.12	0.01	0.12
3	0.99	0.01	0.16	0.50	0.99	0.08
4	0.99	0.01	0.34	0.01	0.01	0.01
5	0.01	0.20	0.47	0.06	0.01	0.33
6	0.01	0.12	0.66	0.12	0.01	0.02

7	0.99	0.61	0.43	0.12	0.99	0.03
8	0.99	0.03	0.82	0.34	0.01	0.16
9	0.01	0.37	0.99	0.01	0.99	0.17
10	0.01	0.16	0.84	0.34	0.99	0.08
11	0.99	0.48	0.72	0.28	0.50	0.09
12	0.99	0.39	0.68	0.39	0.99	0.12

Tabel 4.17 Data Y_{uji}

Y_{uji}	1
1	1
2	1
3	1
4	1
5	2
6	2
7	2
8	2
9	3
10	3
11	3
12	3

4.4.3.1 Proses Hitung H_{init}

Setelah mendapatkan nilai bobot masukan dan target selanjutnya matriks bobot akan ditranspose dan kemudian akan ditambah dengan *ones* dari bias. Hasil proses hitung matriks H_{init} dapat dilihat pada Tabel 4.18 dan contoh perhitungan akan titampilkan dibawah.

$$H_{init(1,1)} = X_{testing}.W^T + b$$

= ((0.69*-0.25) + (0.76*0.31) + (0.38*-0.21)) + 0.44

= 0.423

Tabel 4.18 Tabel Keluaran H_{init}

H_{init}	1	2	3
1	0.65	0.39	0.77
2	0.61	0.48	0.56
3	1.50	1.01	1.28
4	0.86	0.64	0.81
5	0.76	0.52	0.55
6	0.72	0.56	0.55

7	1.89	1.56	1.69
8	1.51	1.03	1.35
9	1.75	1.41	1.06
10	1.58	1.18	1.03
11	1.90	1.48	1.77
12	2.14	1.62	1.83

4.4.3.2 Proses Hitung Matriks *H*

Proses yang dilakukan setelah mendapatkan nilai H_{init} adalah proses menghitung nilai *H.* Contoh proses perhitungan nilai *H* dengan fungsi aktivasi akan ditampilkan pada Tabel 4.19 dan contoh perhitungannya juga akan ditampilkan dibawah.

$$H = \frac{1}{1+\exp(-(X.W^T)+b)}$$

$$H = \frac{1}{1+\exp(-H_{init})}$$

$$H_{(1,1)} = \frac{1}{1+\exp(-0.423)}$$

$$= 0.604$$

Tabel 4.19 Hasil Perhitungan *H*

H	1	2	3
1	0.66	0.60	0.68
2	0.65	0.62	0.64
3	0.82	0.73	0.78
4	0.70	0.65	0.69
5	0.68	0.63	0.63
6	0.67	0.64	0.63
7	0.87	0.83	0.84
8	0.82	0.74	0.79
9	0.85	0.80	0.74
10	0.83	0.77	0.74
11	0.87	0.81	0.85
12	0.90	0.83	0.86

4.4.3.3 Proses Hitung Nilai $\hat{Y}$

Proses yang dilakukan setelah mendapat nilai *H* adalah menghitung nilai $\hat{Y}$. Pada proses ini nilai *H* yang telah didapat akan dikalikan dengan nilai keluaran $\hat{\beta}$ yang didapat sebelumnya pada proses latih ELM. Hasil perhitungan akan ditampilkan pada Tabel 4.20 berikut dengan contoh perhitungannya.

$$\hat{Y} = H.\hat{\beta}$$

$$\hat{Y}(1,1) = (0.604*-16.55) + (0.649*-8.11) + (0.760*20.49)$$

$$\hat{Y}(1,1) = 0.300$$

Tabel 4.20 Hasil Perhitungan Nilai $\hat{Y}$

Prediksi $\hat{Y}$	1
1	0.79
2	2.17
3	1.56
4	1.87
5	2.06
6	2.49
7	3.04
8	1.53
9	3.87
10	3.06
11	2.50
12	2.69

4.5 Perancangan Skenario Pengujian

Pengujian yang akan dilakukan pada penelitian ini merupakan pengujian terkait dengan tujuan mengetahui performa dan akurasi pada algoritme *Extreme Machine Learning* yang digunakan untuk klasifikasi penanganan *Human Papilloma Virus*. Skenario pengujian yang dilakukan pada penelitian ini adalah sebagai berikut:

1. Pengujian perbandingan data latih dan data uji
2. Pengujian jumlah *hidden neuron*
3. Pengujian pengaruh *hidden neuron* terhadap waktu proses
4. Pengujian fungsi aktivasi

4.5.1 Pengujian Pengaruh Perbandingan Data Latih dan Data Uji Terhadap Akurasi

Pengujian perbandingan data latih dan data uji dilakukan untuk mengetahui pengaruh dari rasio jumlah data latih dan jumlah data uji terhadap kemampuan algoritme *Extreme Learning Machine* dalam mengenali pola data. Rancangan pengujian untuk perbandingan data latih dan data uji akan ditunjukkan pada Tabel 4.21.

Tabel 4.21 Skernario Pengujian Perbandingan Data Latih dan Data Uji

Percobaan ke-*i*	Nilai Akurasi Pada Perbandingan Data Latih dan Data Uji								
	90:10	80:20	70:30	60:40	50:50	40:60	30:70	20:80	10:90
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
Rata-rata Akurasi									

4.5.2 Pengujian Pengaruh Jumlah *Hidden Neuron* Terhadap Akurasi

Pengujian jumlah *hidden neuron* dilakukan untuk mengetahui pengaruh jumlah *hidden neuron* terhadap nilai akurasi dalam implementasi algoritme ELM. Rentang bobot masukan yang digunakan pada penelitian ini adalah antara -1 sampai 1 dan rentang nilai bias yang digunakan pada pengujianini adalah antara 0 sampai 1. Perbandingan data latih dan data uji menggunakan rasio 70%:30% untuk data latih dan data uji. Rancangan pengujian untuk pengujian jumlah *hidden neuron* akan ditunjukkan pada Tabel 4.22.

Tabel 4.22 Skenario PerngujianPengaruh *Hidden Neuron* Terhadap Akurasi

Percobaan ke-*i*	Nilai Akurasi Pada Jumlah *Hidden Neuron*								
	1	3	5	10	15	30	50	100	200
1									
2									
3									
4									
5									
6									
7									
8									

9									
10									
Akurasi									

4.5.3 Pengujian Pengaruh *Hidden Neuron* Terhadap *Learning Rate*

Pengujian jumlah *hidden neuron* dilakukan untuk mengetahui pengaruh jumlah *hidden neuron* terhadap *learning rate* dalam implementasi algoritme ELM. Rentang bobot masukan yang digunakan pada penelitian ini adalah antara -1 sampai 1 dan rentang nilai bias yang digunakan pada pengujianini adalah antara 0 sampai 1. Perbandingan data latih dan data uji menggunakan rasio 70%:30% untuk data latih dan data uji. Rancangan pengujian untuk pengujian jumlah *hidden neuron* akan ditunjukkan pada Tabel 4.23.

Tabel 4.23 Skenario Pengujian Pengaruh *Hidden Neuron* Terhadap *Learning Rate*

Percobaan ke-*i*	*Learning Rate* Pada Jumlah *Hidden Neuron*								
	1	3	5	10	15	30	50	100	200
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
Akurasi									

4.5.4 Pengujian Pengaruh Fungsi Aktivasi Terhadap Akurasi

Pada pengujian ini akan dilakukan uji terhadap pengeruh penggunaan fungsi aktivasi pada akurasi yang akan didapatkan. Perbandingan data latih dan data uji menggunakan rasio 70%:30% untuk data latih dan data uji dan *hidden neuron* yang digunakan berjumlah 10. Rancangan pengujian untuk penggunaan fungsi aktivasi akan ditunjukkan pada Tabel 4.24.

Tabel 4.24 SkenarioPengaruh Fungsi Aktivasi Terhadap Akurasi

Percobaan ke-*i*	Nilai Akurasi Pada Fungsi Aktivasi				
	Sigmoid Biner	*Linear*	*Sigmoid Bipolar*	*Sin*	*Radial Basis*
1					
2					
3					

4					
5					
6					
7					
8					
9					
10					
Akurasi					

BAB 5 IMPLEMENTASI

5.1 Implementasi Sistem

Implementasi dilakukan berdasarkan perancangan yang telah dilakukan pada bab sebelumnya. Pada pengimplementasian pada penelitian ini menggunakan bahasa pemrograman C# pada Microsoft Visual Studio 2010.

5.1.1 Implementasi Normalisasi Data

Metode yang digunakan dalam normalisasi data pada penelitian ini adalah normalisasi Min-Max. Kode program proses normalisasi data ditunjukkan pada Tabel 5.1.

Tabel 5.1 Kode Program Normalisasi Data

No	Kode Program
1	`public double[,] normalisasi(double[,] data_normalisasi)`
2	`{`
3	`data_normalisasi = readDataGrid();`
4	`double BA = 0.99;`
5	`double BB = 0.01;`
6	`double minSex = 1; double minAge = 15; double minTime = 0.25;`
7	`double minNoW = 1; double minType = 1; double minArea = 4;`
8	`double maxSex = 2; double maxAge = 67; double maxTime = 12;`
9	`double maxNoW = 19; double maxType = 3; double maxArea = 900;`
10	`int Sex = 0; int Age = 1; int Time = 2; int NoW = 3; int Type = 4;`
11	`int Area = 5;`
12	`for (int z = 0; z < data_normalisasi.GetLength(0); z++)`
13	`{`
14	`data_normalisasi[z, Sex] = (BA - BB) * ((data_normalisasi[z, Sex]`
15	`- minSex) / (maxSex - minSex)) + BB;`
16	`}`
17	`for (int z = 0; z < data_normalisasi.GetLength(0); z++)`
18	`{`
19	`data_normalisasi[z, Age] = (BA - BB) * ((data_normalisasi[z, Age]`
20	`- minAge) / (maxAge - minAge)) + BB;`
21	`}`
22	`for (int z = 0; z < data_normalisasi.GetLength(0); z++)`
23	`{`
24	`data_normalisasi[z, Time] = (BA - BB) * ((data_normalisasi[z,`
25	`Time] - minTime) / (maxTime - minTime)) + BB;`
26	`}`
27	`for (int z = 0; z < data_normalisasi.GetLength(0); z++)`
28	`{`
29	`data_normalisasi[z, NoW] = (BA - BB) * ((data_normalisasi[z,`
30	`NoW] - minNoW) / (maxNoW - minNoW)) + BB;`
31	`}`
32	`for (int z = 0; z < data_normalisasi.GetLength(0); z++)`
33	`{`
34	`data_normalisasi[z, Type] = (BA - BB) * ((data_normalisasi[z,`
35	`Type] - minType) / (maxType - minType)) + BB;`
36	`}`
37	`for (int z = 0; z < data_normalisasi.GetLength(0); z++)`
38	`{`

```
39          data_normalisasi[z, Area] = (BA - BB) * ((data_normalisasi[z,
40  Area] - minArea) / (maxArea - minArea)) + BB;
41      }
42    return data_normalisasi;
43  }
```

Berikut ini merupakan penjelasan kode program untuk normalisasi data yang digunakan pada penelitian ini:

Baris 3 s.d 13 : Inisialisasi data_normalisasi, batas atas, batas bawah, dan indeks kolom fitur *Sex*, *Age*, *Type*, *Number of Warts*, *Area*, *Time*.

Baris 14 s.d 18 : Proses perulangan untuk normalisasi fitur *Sex*.

Baris 19 s.d 23 : Proses perulangan untuk normalisasi fitur *Age*.

Baris 24 s.d 28 : Proses perulangan untuk normalisasi fitur *Time*.

Baris 29 s.d 33 : Proses perulangan untuk normalisasi fitur *Number of Warts*.

Baris 34 s.d 38 : Proses perulangan untuk normalisasi fitur *Type*.

Baris 39 s.d 43 : Proses perulangan untuk normalisasi *Area*.

Baris 44 : Mengembalikan nilai data_normalisasi yang telah dinormalisasi.

5.1.2 Implementasi Inisalisasi Bobot Masukan

Nilai yang digunakan pada bobot masukan berupa nilai acak dengan rentang nilai -1 sampai dengan 1. Pada Tabel 5.2 ditampilkan kode program untuk proses inisialisasi bobot masukan.

Tabel 5.2 Kode Program Inisialisasi Bobot Masukan

No	Kode Program
1	`public double[,] generate_bobot(int hidden_neuron, int input_layer)`
2	`{`
3	`  double[,] bobot_input = new double[hidden_neuron, input_layer];`
4	`  Random random = new Random();`
5	`  double input;`
6	`  for (int i = 0; i < bobot_input.GetLength(0); i++)`
7	`    {`
8	`       for (int j = 0; j < bobot_input.GetLength(1); j++)`
9	`     {`
10	`        input = random.NextDouble() * (0.99 - -0.99) + -0.99;`
11	`        bobot_input[i, j] = System.Math.Round(input, 2);`
12	`     }`
13	`    }`
14	`    return bobot_input;`
15	`  }`

Berikut ini merupakan penjelasan kode program untuk proses inisialisasi bobot masukan yang digunakan pada penelitian ini:

Baris 1 : Pada fungsi generate_bobot akan mendapatkan parameter masukan berupa hidden_neuron dan input_layer.

Baris 3 : Pembuatan matriks bobot_input dengan *ordo hidden neuron* x *input layer*.

Baris 4 s.d 5 : Inisialisasi objek acak dan objek nilai bobot.

Baris 6 s.d 13 : Perulangan untuk mendapatkan nilai matriks bobot_input dengan nilai acak antara -1 sampai 1 dan pembulatan nilai dua angka dibelakang koma.

Baris 14 : Mengembalikan nilai matriks bobot_input.

5.1.3 Implementasi Inisialisasi Bias

Nilai yang digunakan pada bias berupa nilai acak dengan rentang nilai 0 sampai dengan 1 yang menghasilkan martiks berordo 1 x X. Pada Tabel 5.3 ditampilkan kode program untuk proses inisialisasi bias.

Tabel 5.3 Kode Program Inisialisasi Bias

No	Kode Program
1	`public double[,] generate_bias(int hidden_neuron)`
2	`{`
3	`  double[,] bias = new double[1, hidden_neuron];`
4	`  Random rand = new Random();`
5	`  double masukan;`
6	`  for (int i = 0; i < hidden_neuron; i++)`
7	`  {`
8	`    masukan = rand.NextDouble() * (0.99 – 0.01) + 0.01;`
9	`    bias[0, i] = Math.Abs(Math.Round(masukan, 2));`
10	`  }`
11	`  return bias;`
12	`}`

Berikut ini merupakan penjelasan kode program untuk inisialisasi bias yang digunakan pada penelitian ini:

Baris 1 : Fungsi generate_bias menerima parameter masukan hidden_neuron.

Baris 3 : Proses pembentukan matriks bias dengan *ordo* 1 x *hidden neuron*.

Baris 4 s.d 5 : Inisialisasi objek *random* dan nilai masukan.

Baris 6 s.d 10 : Proses perulangan untuk mengisi matriks bias dengan nilai acak antara 0 sampai 1 dan pembulatan dua angka dibelakang koma.

Baris 11 : Mengembalikan nilai matriks bias.

5.1.4 Implementasi Transpose Matriks

Proses transpose metriks bertujuan untuk mengubah baris dan kolom awal menjadi kolom dan baris pada metriks baru Pada Tabel 5.4 ditampilkan kode program untuk transpose matriks.

Tabel 5.4 Kode Program Transpose Matriks

No	Kode Program
1	`public double[,] matriks_transpose(double[,] matriks)`
2	`{`
3	`  double[,]matriks_transpose=new double[matriks.GetLength(1),`
4	`matriks.GetLength(0)];`
5	
6	`  for (int i = 0; i < matriks_transpose.GetLength(0); i++)`
7	`  {`
8	`   for (int j = 0; j < matriks_transpose.GetLength(1); j++)`
9	`   {`
10	`     matriks_transpose[i, j] = matriks[j, i];`
11	`   }`
12	`  }`
13	`  return matriks_transpose;`
14	`}`

Berikut ini merupakan penjelasan kode program untuk transpose matriks yang digunakan pada penelitian ini:

Baris 1 :Fungsi matriks_transpose menerima parameter matriks masukan sebagai matriks.

Baris 3 :Proses pembentukan matriks baru dengan nama matriks_transpose dengan *ordo* kolom matriks masukan x baris matriks masukan.

Baris 6 s.d 12 : Proses perulangan untuk melakukan transpose matriks dengan cara mengubah baris dan kolom matriks masukan menjadi kolom dan baris baru yang disimpan pada matriks baru matriks_transpose.

Baris 13 : Mengembalikan nilai matriks_transpose.

5.1.5 Implementasi Perkalian Matriks

Proses perkalian matriks bertujuan untuk mengalikan dua buah matriks dan menghasilkan sebuah matriks baru. Pada Tabel 5.5 ditampilkan kode program untuk proses perkalian matriks.

Tabel 5.5 Kode Program Perkalian Matriks

No	Kode Program
1	`public`
2	`double[,]matriks_perkalian(double[,]MatriksA,double[,]MatriksB)`
3	`{`
4	`  double[,] matriks_hasil = new double[MatriksA.GetLength(0),`
5	`MatriksB.GetLength(1)];`
6	`  for (int i = 0; i < MatriksA.GetLength(0); i++)`
7	`  {`
8	`   for (int j = 0; j < MatriksB.GetLength(1); j++)`

```
 9    {
10     matriks_hasil[i, j] = 0;
11     for (int k = 0; k < MatriksA.GetLength(1); k++)
12     {
13
14  matriks_hasil[i,j]=matriks_hasil[i,j]+MatriksA[i,k]*MatriksB[k,j];
15      matriks_hasil[i, j] = Math.Round(matriks_hasil[i, j], 4);
16     }
17    }
18   }
19   return matriks_hasil;
20  }
```

Berikut ini merupakan penjelasan kode program untuk proses perkalian matriks yang digunakan pada penelitian ini:

Baris 1 : Fungsi matriks_perkalian menerima parameter sebagai matriksA dan matriksB.

Baris 3 :Proses pembentukan matriks baru dengan nama matriks_hasil dengan *ordo* baris matriksA x kolom matriksB.

Baris 5 s.d 18 : Proses perulangan untuk melakukan perkalian matriks dengan cara mengalikan baris matriksA dengan kolom matriksB dan menyimpan hasil perkalian matriks pada matriks_hasil.

Baris 19 : Mengembalikan nilai matriks_hasil.

5.1.6 Implementasi Penjumlahan Matriks dengan Bias

Untuk melakuakn perhitungan H_{init} diperlukan masukan berupa hasil penjumlahan matriks bobot masukan dan matriks bias. Pada Tabel 5.6 ditampolkan kode program pada proses penjumlahan matriks bobt masukan dengan matriks bias.

Tabel 5.6 Kode Program Penjumlahan Matriks Bobot Masukan dengan Matriks Bias

No	Kode Program
1	`public double[,] ones_bias(double[,] H_init, double[,] bias)`
2	`{`
3	`for (int i = 0; i < H_init.GetLength(0); i++ )`
4	`{`
5	`for (int j = 0; j < H_init.GetLength(1); j++)`
6	`{`
7	`H_init[i, j] = H_init[i, j] + bias[0, j];`
8	`}`
9	`}`
10	`return H_init;`
11	`}`

Berikut ini merupakan penjelasan kode program untuk proses penjumlahan bobot masukan dengan bias yang digunakan pada penelitian ini:

Baris 1 : Fungsi ones_bias menerima parameter sebagai matriks H_init dan matriks bias.

Baris 3 s.d 9 : Proses perulangan untuk melakukan penjumlahan matriks H_init dengan matriks bias sehingga diperoleh matriks H_init.

Baris 10 : Mengembalikan nilai matriks H_init.

5.1.7 Impkementasi Proses Latih

5.1.7.1 Implementasi Hitung H_{init}

Pada tahapan implementasi ini bertujuan untuk menghitung nilai matriks H_{init} dengan melakukan perkalian matriks X_{latih} dengan matriks bobot masukan W^T dan hasil perkalian tersebut akan ditambah dengan nilaimatriks bias. Implementasi perhitungan H_{init} akan ditunjukkan pada Tabel 5.7.

Tabel 5.7 Kode Program Perhitungan H_{init}

No	Kode Program
1	`bobot_input_transpose = matriks_transpose(bobot_input);`
2	`H_init_uji = matriks_perkalian(X_uji, bobot_input_transpose);`
3	`H_init_uji = ones_bias(H_init_uji, bias);`

Berikut ini merupakan penjelasan kode program untuk proses perhitungan H_{init} yang digunakan pada penelitian ini:

Baris 2 : Melakukan proses transpose matriks bobot dengan cara melakukan pemanggilan fungsi matriks_transpose dan memberikan matriks bobot_input sebagai matriks masukan untuk fungsi tersebut serta menerima kembalian fungsi transpose sebagai matriks bobot_input_transpose. Fungsi matriks_transpose telah dijabarkan pada Tabel 5.4.

Baris 3 : Melakukan proses perkalian matriks dengan cara melakukan pemanggilan fungsi matriks_perkalian dan memberikan matriks X_training dan bobot_input_transpose sebagai matriks masukan untuk fungsi tersebut serta menerima kembalian fungsi perkalian sebagai matriks H_init. Fungsi matriks_perkalian telah dijabarkan pada Tabel 5.5.

Baris 4 : Melakukan proses penjumlahan matriks dengan cara melakukan pemanggilan fungsi ones_bias dan memberikan matriks masukan H_init dan bias serta menerima kembalian fungsi sebagai matriks H_init. Fungsi ones_bias telah dijabarkan pada Tabel 5.6.

5.1.7.2 Implementasi Hitung Nilai *H*

Hitungan *H* merupakan proses untuk menghitung matriks keluaran *hidden layer* yakni matriks *H* dengan menggunakan fungsi aktivasi *sigmoid biner*. Pada Tabel 5.8 akan ditampilkan kode program untuk implementasi hitung nilai *H.*

Tabel 5.8 Kode Program Hitung *H*

No	Kode Program
1	`H = sigmoid_biner(H_init, H);`
2	
3	`public double[,] sigmoid_biner(double[,] H_init, double[,] H)`
4	`{`
5	`for (int i = 0; i < H_init.GetLength(0); i++ )`
6	`{`
7	`for (int j = 0; j < H_init.GetLength(1); j++)`
8	`{`
9	`H[i, j] = 1 / (1 + Math.Exp(-H_init[i, j]));`
10	`}`
11	`}`
12	`return H;`
13	`}`

Berikut ini merupakan penjelasan kode program untuk proses perhitungan *H* yang digunakan pada penelitian ini:

Baris 2 : Melakukan proses perhitungan H dengan cara memanggil fungsi sigmoid_biner. Fungsi sigmoid_biner membutuhkan parameter masukan yaitu matriks H_init yang telah dihitung sebelumnya. Fungsi sigmoid_biner akan memberikan nilai kembalian yang akan disimpan pada matriks H_uji.

Baris 4 : Fungsi sigmoid_biner menerima parameter sebagai matriks H_init dan matriks H.

Baris 6 s.d 12 : Proses perulangan untuk melakukan perhitungan H menggunakan fungsi aktivasi *sigmoid biner*.

Baris 13 : Mengembalikan nilai matriks H.

5.1.7.3 Implemantasi Hitung Inverse Matriks

Hitung invers matriks merupakan proses yang dilakukan untuk invers matriks menggunakan Operasi Baris Elementer (OBE). Pada Tabel 5.9 akan ditampilkan kode program untuk implementasi hitung nilai inverse matriks.

Tabel 5.9 Kode Program Inverse Matriks

No	Kode Program
1	`public double[,] inverse_matriks(double[,] matriksA)`
2	`{`
3	`double[,]matriks=new double[matriksA.GetLength(0),`
4	`matriksA.GetLength(1)];`
5	`double[,] matriks_invers = new double[matriks.GetLength(0),`
6	`matriks.GetLength(1)];`
7	
8	`for (int i = 0; i < matriks.GetLength(0); i++)`

```
 9   {
10    for (int j = 0; j < matriks.GetLength(1); j++)
11    {
12     matriks[i, j] = matriksA[i, j];
13    }
14   }
15
16   for (int i = 0; i < matriks.GetLength(0); i++)
17   {
18    for (int j = 0; j < matriks.GetLength(1); j++)
19    {
20     if (i == j)
21     {
22      matriks_invers[i, j] = 1;
23     }
24     else
25     {
26      matriks_invers[i, j] = 0;
27     }
28    }
29   }
30
31   for (int i = 0; i < matriks.GetLength(0); i++)
32   {
33    double t = matriks[i, i];
34    for (int k = 0; k < matriks.GetLength(1); k++)
35    {
36     matriks_invers[i, k] = matriks_invers[i, k] / t;
37     matriks[i, k] = matriks[i, k] / t;
38    }
39
40    for (int j = 0; j < matriks.GetLength(1); j++)
41    {
42     double c = matriks[j, i];
43     for (int l = 0; l < matriks.GetLength(1); l++)
44     {
45      if (i != j)
46      {
47     matriks_invers[j,l]=matriks_invers[j,l] - c*matriks_invers[i, l];
48       matriks[j, l] = matriks[j, l] - c * matriks[i, l];
49      }
50     }
51    }
52   }
53   return matriks_invers;
54  }
```

Berikut ini merupakan penjelasan kode program untuk proses perhitungan inverse matriks yang digunakan pada penelitian ini:

Baris 1 : Fungsi inverse_matriks menerima parameter sebagai matriksA yang akan dilakukan proses invers.

Baris 3 s.d 5 :Inisialisasi matriks baru yaitu *matriks* dan matriks_invers. *Matriks* adalah matriks masukan yang akan diinvers sedangkan matriks_invers adalah matriks yang nantinya bernilai hasil proses invers.

Baris 8 s.d 14 : Proses perulangan untuk menyalin nilai matriks masukan matriksA ke matriks *matriks* agar proses invers tidak menghilangkan nilai matriks awal.

Baris 16 s.d 29 : Proses perulangan untuk membuat matriks_invers menjadi matriks identitas.

Baris 31 s.d 52 : Proses perulangan yang merupakan inti dari proses invers dalam fungsi ini yaitu menyelesaikan proses invers menggunakan metode OBE. Metode OBE menyandingkan matriks awal dengan matriks identitas selanjutnya melakukan serangkaian tahap perhitungan untuk menjadikan matriks awal menjadi matriks identitas sedangkan matriks identitas awal menjadi hasil proses invers.

Baris 53 : Proses mengembalikan nilai matriks_invers.

5.1.7.4 Proses Hitung Matriks H^+

Hitung matriks H^+ atau matriks *Moore Penrose Pseudo Invers* adalah proses untuk menghitung nilai matriks H^+ yang akan digunakan pada proses perhitungan nilai $\hat{\beta}$. Pada Tabel 5.10 akan ditampilkan kode program untuk implementasi hitung nilai matriks H^+.

Tabel 5.10 Kode Program Hitung H^+

No	Kode Program
1	`H_transpose = matriks_transpose(H);`
2	`_HT_H = matriks_perkalian(H_transpose, H);`
3	`HT_H = inverse_matriks(_HT_H);`
4	`H_plus = matriks_perkalian(HT_H, H_transpose);`

Berikut ini merupakan penjelasan kode program untuk proses perhitungan H^+ yang digunakan pada penelitian ini:

Baris 1 : Melakukan proses transpose matriks H dengan cara memanggil fungsi matriks_transpose dan memberikan nilai matriks H sebagai parameter masukan. Fungsi tersebut akan memberikan nilai kembalian dan disimpan pada matriks H_transpose. Fungsi matriks_transpose telah dijabarkan pada Tabel 5.4.

Baris 2 : Melakukan proses perkalian matriks antara matriks H_transpose dengan matriks H menggunakan fungsi matriks_perkalian. Fungsi tersebut akan memberikan nilai kembalian dan disimpan pada matriks _HT_H. Fungsi matriks_perkalian telah ditunjukkan pada Tabel 5.5.

Baris 3 : Melakukan proses invers matriks _HT_H dengan cara memanggil fungsi inverse_matriks. Fungsi tersebut akan memberikan nilai kembalian dan disimpan pada matriks

HT_H. Fungsi inverse_matriks telah ditunjukkan pada Tabel 5.9.

Baris 4 : Melakukan proses perkalian matriks antara matriks HT_H dengan matriks H_transpose menggunakan fungsi matriks_perkalian. Fungsi tersebut akan memberikan nilai kembalian dan disimpan pada matriks H_plus. Fungsi matriks_perkalian telah ditunjukkan pada Tabel 5.5.

5.1.7.5 Implementasi Hitung $\widehat{\boldsymbol{\beta}}$

Hitung $\hat{\beta}$ merupakan untuk mendapatkan nilai $\hat{\beta}$ dengan cara mengalikan matriks H^{+} dengan matriks $Y_{training}$. Pada Tabel 5.11 akan ditampilkan kode program untuk implementasi hitung nilai $\hat{\beta}$.

Tabel 5.11 Kode Program Hitung $\hat{\beta}$

No	Kode Program
1	`beta = matriks_perkalian(H_plus, Y_training);`

Berikut ini merupakan penjelasan kode program untuk proses hitung nilai $\hat{\beta}$ yang digunakan pada penelitian ini:

Baris 1 : Melakukan proses perkalian matriks antara matriks H_plus dengan matriks Y_training menggunakan fungsi matriks_perkalian. Fungsi tersebut akan memberikan nilai kembalian dan disimpan pada matriks beta. Matriks beta akan digunakan untuk tahap uji selanjutnya. Fungsi matriks_perkalian telah ditunjukkan pada Tabel 5.5.

5.1.8 Implementasi Uji *Extreme Learning Machine*

5.1.8.1 Implementasi Hitung H_{init}

Hitung H_{init} merupakan proses untuk menghitung matriks H_{init} dengan cara mengalikan matriks X_{uji} dengan matriks bobot masukan W^{T} dan menambahkan dengan matriks bias. Pada Tabel 5.12 akan ditampilkan kode program untuk implementasi hitung nilai $\hat{\beta}$.

Tabel 5.12 Kode Program Hitung H_{init}

No	Kode Program
1	`bobot_input_transpose = matriks_transpose(bobot_input);`
2	`H_init_uji = matriks_perkalian(X_uji, bobot_input_transpose);`
3	`H_init_uji = ones_bias(H_init_uji, bias);`

Berikut ini merupakan penjelasan kode program untuk proses hitung nilai H_{init} yang digunakan pada penelitian ini:

Baris 1 : Melakukan proses transpose matriks bobot dengan cara melakukan pemanggilan fungsi matriks_transpose dan memberikan matriks bobot_input sebagai matriks masukan untuk fungsi tersebut serta menerima

kembalian fungsi transpose sebagai matriks bobot_input_transpose. Fungsi matriks_transpose telah dijabarkan pada Tabel 5.4.

Baris 2 : Melakukan proses perkalian matriks dengan cara melakukan pemanggilan fungsi matriks_perkalian dan memberikan matriks X_uji dan bobot_input_transpose sebagai matriks masukan untuk fungsi tersebut serta menerima kembalian fungsi perkalian sebagai matriks H_init_uji. Fungsi matriks_perkalian telah dijabarkan pada Tabel 5.5.

Baris 3 : Melakukan proses penjumlahan matriks dengan cara melakukan pemanggilan fungsi ones_bias dan memberikan matriks masukan H_init_uji dan bias serta menerima kembalian fungsi sebagai matriks H_init_uji. Fungsi ones_bias telah dijabarkan pada Tabel 5.6

5.1.8.2 Implementasi Hitung Nilai *H*

Hitung *H* merupakan proses untuk menghitung matriks keluaran *hidden layer* yakni matriks *H* menggunakan fungsi aktivasi *sigmoid biner*. Tabel 5.13 akan ditampilkan kode program untuk implementasi hitung nilai *H*.

Tabel 5.13 Kode Program Hitung Nilai *H*

No	Kode Program
1	`H_uji = sigmoid_biner(H_init_uji, H_uji);`
2	
3	`public double[,] sigmoid_biner(double[,] H_init, double[,] H)`
4	`{`
5	`  for (int i = 0; i < H_init.GetLength(0); i++ )`
6	`  {`
7	`   for (int j = 0; j < H_init.GetLength(1); j++)`
8	`   {`
9	`     H[i, j] = 1 / (1 + Math.Exp(-H_init[i, j]));`
10	`   }`
11	`  }`
12	`  return H;`
13	`}`

Berikut ini merupakan penjelasan kode program untuk proses hitung nilai H_i yang digunakan pada penelitian ini:

Baris 1 : Melakukan proses perhitungan H_uji dengan cara memanggil fungsi sigmoid_biner. Fungsi sigmoid_biner membutuhkan parameter masukan yaitu matriks H_init_uji yang telah dihitung sebelumnya. Fungsi sigmoid_biner akan memberikan nilai kembalian yang akan disimpan pada matriks H_uji.

Baris 3 : Fungsi sigmoid_biner menerima parameter sebagai matriks H_init dan matriks H.

Baris 5 s.d 11 : Proses perulangan untuk melakukan perhitungan H menggunakan fungsi aktivasi *sigmoid biner*.

Baris 12 : Mengembalikan nilai matriks H.

5.1.8.3 Implementasi Hitung Nilai $\hat{Y}$

Hitung $\hat{Y}$ merupakan proses yang bertujuan untuk melakukan perhitungan matriks *Y* prediksi $\hat{Y}$ dengan cara mengalikan matriks *H* dari tahap *uji* dengan matriks $\hat{\beta}$ dari tahap *training* Tabel 5.14 akan ditampilkan kode program untuk implementasi hitung nilai $\hat{Y}$.

Tabel 5.14 Kode Program Hitung Nila*i* $\hat{Y}$

No	Kode Program
1	`Y_prediksi = matriks_perkalian(H_uji, beta);`

Berikut ini merupakan penjelasan kode program untuk proses hitung nilai $\hat{Y}$ yang digunakan pada penelitian ini:

Baris 1 : Melakukan proses perkalian matriks antara matriks H_uji dengan matriks beta menggunakan fungsi matriks_perkalian. Fungsi tersebut akan memberikan nilai kembalian dan disimpan pada matriks Y_prediksi.

5.1.8.4 Implementasi Hitung Akurasi

Hitung nilai akurasi bertujuan untuk melakukan perhitungan akurasi dengan cara membandingkan nilai hasil prediksi klasifikasi dengan data uji yang ada. Pada implementasi pada perhitungan akurasi akan menghasilkan matriks akurasi. Tabel 5.15 akan ditampilkan kode program untuk implementasi hitung nilai akurasi.

Tabel 5.15 Kode Program Hitung Akurasi

No	Kode Program
1	`akurasi = hitung_akurasi(Y_uji, Kelas_prediksi);`
2	
3	`private double hitung_akurasi(double[,] kelas_training, int[,]`
4	`kelas_prediksi)`
5	`{`
6	`  double akurasi;`
7	`  int benar = 0;`
8	`  for (int i = 0; i < kelas_prediksi.GetLength(0); i++)`
9	`  {`
10	`    if ((int) kelas_training[i, 0] == kelas_prediksi[i, 0])`
11	`      {`
12	`         benar++;`
13	`      }`
14	`  }`
15	`   akurasi = (double) benar / kelas_training.GetLength(0);`
16	`   return akurasi;`
17	`}`

Berikut ini merupakan penjelasan kode program untuk proses hitung nilai akurasi yang digunakan pada penelitian ini:

Baris 2 :Melakukan proses perhitungan akurasi dengan cara memanggil fungsi hitung_akurasi. Fungsi hitung_akurasi

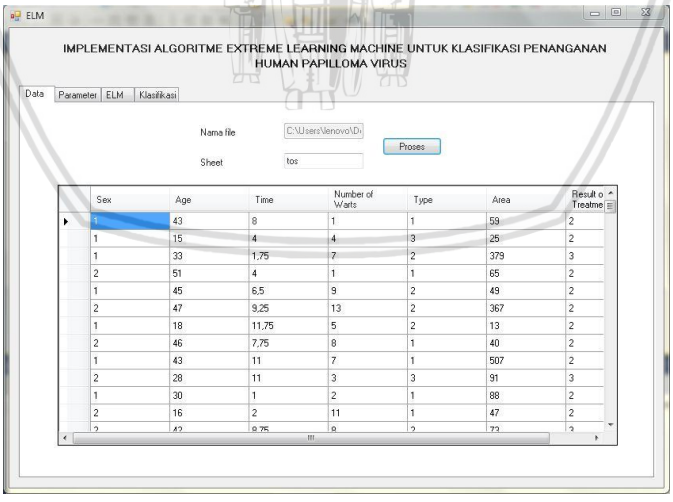
membutuhkan parameter masukan yaitu matriks kelas-prediksi yang telah dihitung sebelumnya, Y_uji dan matriks akurasi. Fungsi hitung_akursi akan memberikan nilai kembalian yang akan disimpan pada matriks akurasi.

Baris 4 s.d 5	:Fungsi hitung_akurasi menerima parameter sebagai matriks kelas_trainingdan matriks kelas_prediksi.
Baris 6 s.d 7	: Inisialisasi variabel akurasi dan variabel benar..
Baris 8 s.d 14	:Proses perulangan untuk melakukan perhitungan akurasi menggunakan rumus akurasi seperti yang telah dijelaskan pada persamaan 2.1.
Baris 15	:Proses perhitungan akurasi dengan membagi nilai variabel benar dibagi dengan kelas_training.
Baris 16	:Mengembalikan nilai akurasi.

5.2 Implementasi Antarmuka

5.2.1 Antarmuka Halaman Data

Antarmuka pengguna halaman data berisi halaman untuk memasukkan dataset yang berupa *file Microsoft Excel* (.xls / .xlsx). Selain itu terdapat *form* untuk mengisi nama *sheet* yang terdapat di *file Excel* tersebut. Pada halaman ini menampilkan tabel dataset yang telah dimuat untuk proses ELM. Pada Gamabar 5.1 ditampilkan antarmuka halaman data.

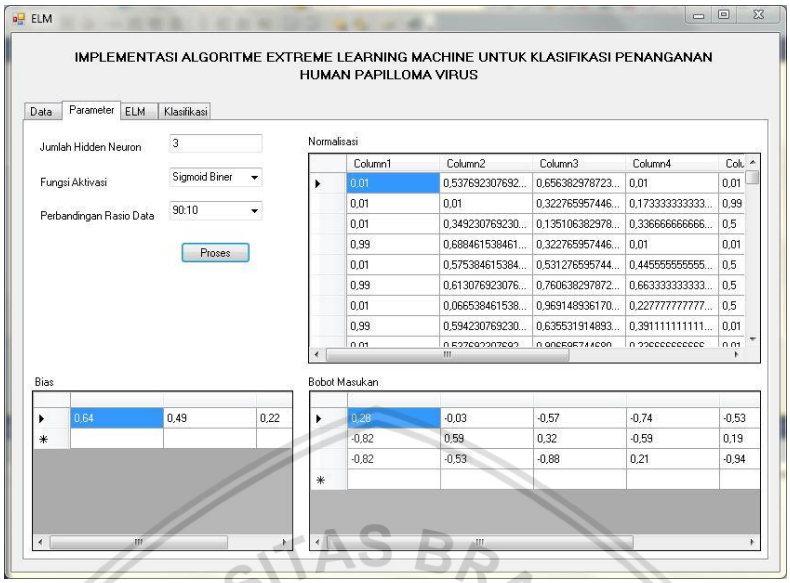


Gambar 5.1 Antarmuka Halaman Data

5.2.2 Antarmuka Halaman Parameter ELM

Antarmuka pengguna halaman parameter ELM berisi halaman untuk memasukkan parameter ELM yaitu jumlah *hidden neuron* dan rasio data latih dan uji. Pada halaman ini akan menampilkan data normalisasi, matriks bobot masukan

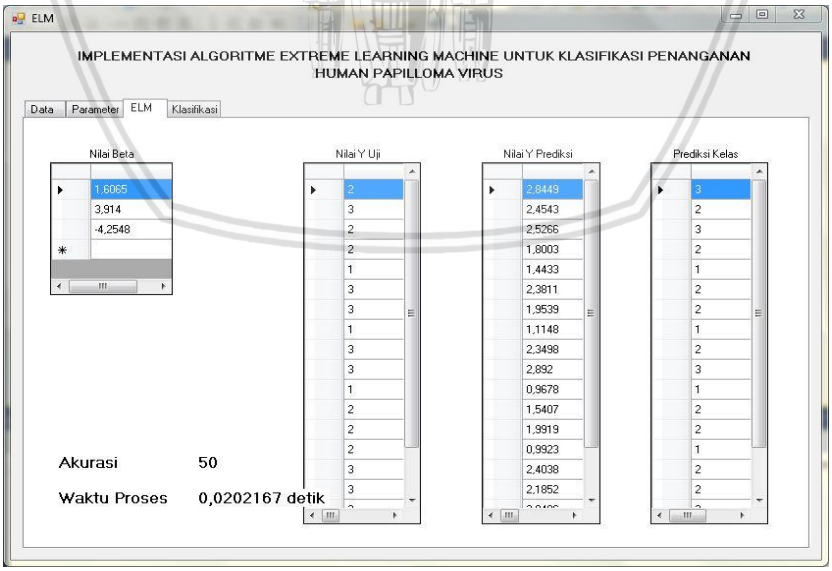
W dan matriks bias setelah *button* Proses ELM ditekan. Pada Gambar 5.2 ditampilkan hasil implementasi antarmuka halaman parameter ELM.



Gambar 5.2 Antarmuka Halaman Parameter ELM

5.2.3 Antarmuka Halaman Hasil ELM

Antarmuka Halaman Hasil ELM berisi halaman yang menampilkan hasil fase *training*, hasil fase *uji* serta nilai akurasi. Pada hasil fase latih akan ditampilkan matriks bobot keluaran $\hat{\beta}$. Sedangkan pada hasil fase uji akan ditampilkan hasil klasifikasi penaganan *Human Papilloma Virus* dari data uji serta matriks Y *uji*. Pada Gambar 5.3 ditampilkan antarmuka halaman hasil ELM



Gambar 5.3 Antermuka Halaman Hasil ELM

BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian

Pada peneliatian ini terdapat beberapa skenario penguijian yang telah dilakukan unutk mengetahui hasil akurasi perhitunagn sistem. Pengujian yang dilakukan pada penelitian ini antara lain :

1. Pengujian pengaruh perbandingan rasio data latih dengan data uji terhadap akurasi.
2. Pengujian pengaruh fungsi aktivasi terhadap akurasi.
3. Pengujian pengaruh jumlah *hidden neuron* terhadap akurasi.
4. Pengujian pengaruh jumlah *hidden neuron* terhadap waktu proses sistem.

Pengujian yang dilakukan pada penelitian ini dilakukan sebanyak 10 kali percobaan pengujian terhadap setiap skenario uji. Berdasarkan dari 10 kali pengujian nantinya akan diketahui rata – rata nilai akurasi dan lama proses eksekusi.

6.2 Hasil dan Analisis Hasil Pengujian

6.2.1 Pengujian Pengaruh Jumlah *Hidden Neuron* Terhadap Akurasi

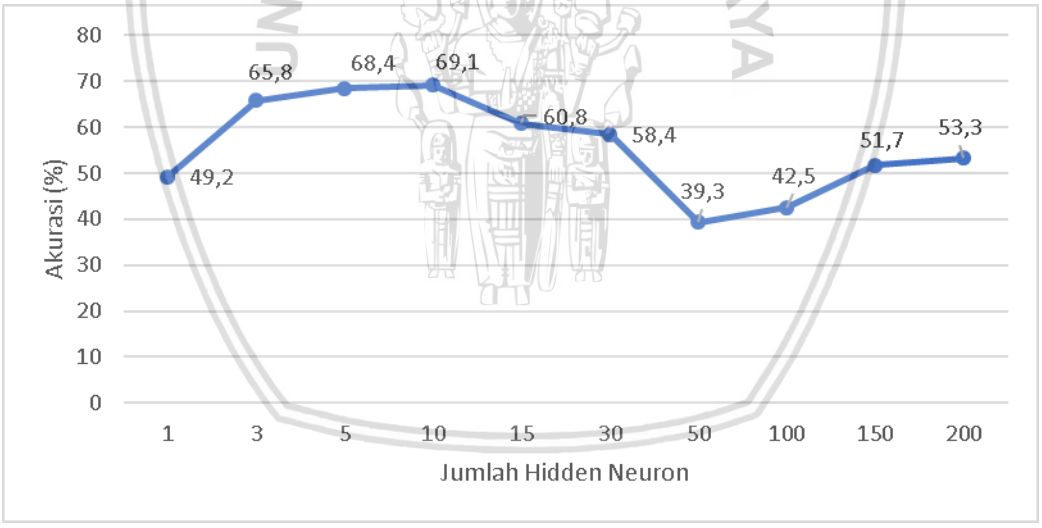
Pengujian ini dilakukan untuk mengetahui pengaruh jumlah *hidden neuron* terhadap hasil akurasi klasifikasi pada implementasi algoritme *Extreme Learning Machine*. Pada pengujian pengaruh jumlah *hidden neuron* terhadap akurasi, *hidden neuron* yang digunakan antara lain 1, 3, 5, 10, 15, 20, 50, 100, 150, dan 200 *hidden neuron.* Rentang bobot masukan yang digunakan adalah antara -1 sampai dengan 1 dan rentang nilai bias antara 0 sampai dengan 1. Selain itu, pada pengujian ini rasio data latih dan uji yang digunakan adalah rasio 90% banding 10% karena pada pengujian pendahuluan rasio data tersebut memiliki nilai rataan akurasi yang paling besar.

Pada hasil implementasi perhitungan ELM yang telah dilakukan nantinya akan menghasilkan nilai akurasi. Nilai akurasi yang dilakukan pada tiap percobaan akan mendapatkan nilai akurasi yang berbeda dikarenakan nilai bobot masukan dan bias yang digunakan pada tiap eksekusi memiliki nilai yang berbeda atau acak dengan rentang nilai yang telah diatur sebelumnya. Oleh sebab itu maka dilakukan 10 kali percobaan agar didapat nilai rataan akurasi. Hasil pengujian perngaruh jumlah *hidden neuron* terhadap akurasi dapat dilihat pada Tabel 6.1.

Tabel 6.1 Pengujian Pengaruh Jumlah *Hidden Neuron* Terhadap Akurasi

Percobaan ke-*i*	Nilai Akurasi Pada Jumlah *Hidden Neuron*									
	1	3	5	10	15	30	50	100	150	200
1	25	58	67	75	67	58	42	42	8	50
2	50	67	67	67	75	58	25	33	67	58
3	42	67	67	58	67	58	42	67	58	67
4	75	75	67	75	58	67	67	25	75	58
5	42	75	75	58	75	67	25	58	42	50
6	67	58	58	58	58	67	25	50	50	42
7	75	67	50	83	58	42	42	42	67	58
8	25	75	83	67	42	50	33	33	58	50
9	58	58	75	67	50	50	42	17	42	58
10	33	58	75	83	58	67	50	58	50	42
Akurasi	49,2	65,8	68,4	69,1	60,8	58,4	39,3	42,5	51,7	53,3

Berdasarkan hasil pengujian pengaruh jumlah *hidden neuron* terhadap akurasi algoritme ELM yang ditunjukkan pada Tabel 6.1 dapat dilihat bahwa akurasi tertinggi adalah 69,1% dengan jumlah *hidden neuron* sebanyak 10 dan akurasi terendah adalah 39,3% dengan jumlah *hidden neuron* sebanyak 50 buah.



Gambar 6.1 Grafik Hasil Pengujian Pengaruh Jumlah *Hidden Neuron* Terhadap Akurasi

Pada grafik di Gambar 6.1 merupakan perbandingan hasil uji yang mana dapat diambil kesimpulan bahwa semakin banyak jumlah *hidden neuron* yang digunakan pada algoritme ELM tidak selalu membuat nilai akurasi dari sistem semakin baik begitu juga sebaliknya. Pada grafik juga ditunjukkan adanya anomali hasil pengujian dengan perbedaan *hidden neuron* yang digunakan. Hal in terjadi dikarenakan penggunaan bobot masukan dan bias pada setiap proses berbeda dikarenakan bobot masukan dan bias didapat secara acak dengan rentangnilai

antara -1 sampai 1 pada bobot masukan dan rentang nilai antara 0 sampai 1 pada bias.

6.2.2 Pengujian Pengaruh Jumlah *Hidden Neuron* Terhadap Waktu Latih (*Learning Rate*)

Pengujian pengaruh jumlah *hidden neuron* terhadap waktu proses algoritme ELM ditujukan untuk mengetahui seberapa efektifnya algoritme ELM dalam menyelesaikan permasalahan. ELM merupakan algoritme yang ada pada jaringan syaraf tiruan. Pada jaringan syaraf tiruan memiliki beberapa *layer* yang saling terkait. *Hidden layer* atau sering disebut dengan *hidden neuron* berfungsi untuk melakukan transformasi nilai masukan menjadi nilai keluaran.

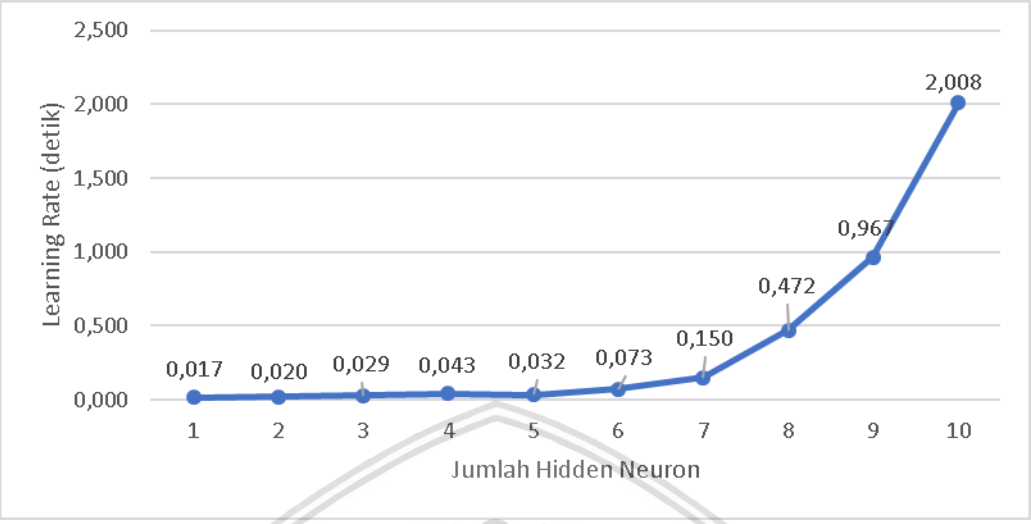
Pengujian pengaruh jumlah *hidden neuron* terhadap waktu proses menggunakan rasio data 80 banding 20 atau dengan menggunakan 80% data latih berbanding 20% data uji. Selain itu pada pengujian ini menggunakan fungsi aktivasi sigmoid biner. Hasil pengujian jumlah *hidden neuron* terhadap waktu proses dapat dilihat pada Tabel 6.2.

Tabel 6.2 Pengujian Pengaruh Jumlah *Hidden Neuron* Terhadap *Learning Rate*

Percobaan ke-*i*	*Learning Rate* Terhadap Jumlah *Hidden Neuron*									
	1	3	5	10	15	30	50	100	150	200
1	0,026	0,033	0,024	0,048	0,052	0,083	0,176	0,649	1,423	2,451
2	0,016	0,013	0,023	0,037	0,026	0,075	0,137	0,431	0,982	1,897
3	0,011	0,023	0,024	0,037	0,032	0,080	0,145	0,436	1,006	1,921
4	0,014	0,017	0,033	0,038	0,013	0,071	0,145	0,441	1,025	1,909
5	0,014	0,015	0,026	0,039	0,032	0,067	0,155	0,433	0,982	1,942
6	0,020	0,019	0,027	0,055	0,033	0,068	0,146	0,475	1,002	1,961
7	0,016	0,018	0,029	0,042	0,036	0,070	0,154	0,479	0,995	2,002
8	0,017	0,022	0,029	0,042	0,032	0,069	0,142	0,480	1,060	1,985
9	0,018	0,019	0,036	0,042	0,033	0,076	0,154	0,447	0,099	2,003
10	0,022	0,020	0,038	0,047	0,034	0,072	0,146	0,448	1,097	2,009
(detik)	0,017	0,020	0,029	0,043	0,032	0,073	0,150	0,472	0,967	2,008

Berdasarkan pengujian pengaruh jumlah *hidden neuron* terhadap lama waktu proses yang tertera pada Tabel 6.2 dapat dilihat bahwa proses elm dengan menggunakan 1 *hidden neuron* merupakan yang tercepat dengan waktu proses

selama 0,017 detik. Sedangakan waktu proses terlama adalah dengan waktu 2,008 detik yang menggunakan *hidden neuron* sebanyak 200.



Gambar 6.2 Penujian Pengaruh Jumlah *Hidden Neuron* Terhadap *Learning Rate*

Pada grafik di Gambar 6.2 merupakan perbandingan hasil uji jumlah *hidden neuron* terhadap *learning rate* yang mana dapat diambil kesimpulan bahwa semakin banyak jumlah *hidden neuron* yang digunakan pada algoritme ELM sebanding dengan lama waktu proses perhitungan dari ELM itu sendiri. Hal ini disebabkan oleh proses perhitungan fungsi aktivasi pada *hidden layer* yang semakin lama pula. Pengujian yang dilakukan dapat membuktikan bahwa banyaknya *hidden neuron* memiliki pengaruh terhadap *learning rate* proses dalam menyelesaikan algoritme ELM.

6.2.3 Pengujian Pengaruh Perbandingan Rasio Data Terhadap Akurasi

Pengujian ini dilakukan untuk mengetahui pengaruh perbandingan rasio data terhadap akurasi sistem yang telah dibuat terhadap kemampuan ELM dalam mengenali pola data yang ada dan akurasi yang dihasilkan. Perbandingan rasio data yang dimaksud adalah perbandingan dari data latih dan data uji yang ada.

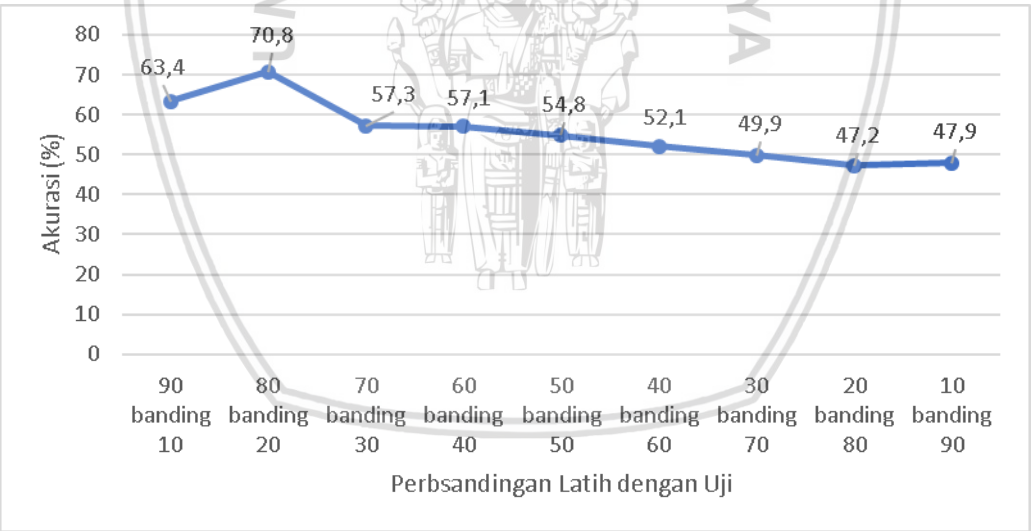
Pada pengujian ini terdapat 9 jenis perbandingan data latih dan data uji yang digunakan yaitu 90%:10%, 80%:20%, 70%:30%, 60%:40%, 50%:50%, 40%:60%, 30%:70%, 20%:80%, dan 10%:90%. Selain dari 9 jenis perbandingan yang ada, pada pengujian ini jumlah *hidden neuron* yang digunakan sebanyak 10 *hidden neuron* dan dengan menggunakan fungsi aktivasi Sigmoid Biner.

Hasil dari pengujian ini akan menghasilkan nilai akurasi yang berbeda beda dikarenakan pada setiap eksekusi nilai bobot masukan dan nilai bias yang digunakan akan didapat secara acak dengan rentang masukan antara -1 sampai dengan 1 dan bias antara 0 sampai dengan 1. Selain itu pada pengujian ini nantinya dilakukan 10 kali percobaan pengujian yang akan menghasilkan rataan nilai akurasu sebagai tujuan utama. Hasil pengujian perngaruh perbandingan rasio data terhadap akurasi sistem dapat dilihat pada Tabel 6.3

Tabel 6.3 Pengujian Pengaruh Perbandingan Rasio Data Terhadap Akurasi

Percobaan ke-*i*	Nilai Akurasi Pada Perbandingan Data Latih dan Data Uji								
	90:10	80:20	70:30	60:40	50:50	40:60	30:70	20:80	10:90
1	50	62	69	68	59	41	47	60	44
2	75	62	71	62	54	58	45	46	52
3	75	71	31	34	44	62	49	54	44
4	67	71	60	53	56	48	29	45	59
5	75	75	57	55	39	56	51	39	56
6	67	71	54	60	61	55	59	43	66
7	58	79	51	60	63	55	61	62	56
8	50	71	60	66	61	52	51	43	42
9	75	67	57	51	53	42	47	41	20
10	42	79	63	62	58	52	60	39	40
Rata-rata Akurasi	63,4	70,8	57,3	57,1	54,8	52,1	49,9	47,2	47,9

Berdasarkan pengujian pengaruh perbandingan rasio data terhadap akurasi yang tertera pada Tabel 6.3 dapat dilihat bahwa proses ELM dengan menggunakan perbandingan rasio data 80% data latih banding 20% data uji memiliki akurasi tertinggi dengan nilai rataan akurasi sebesar 70,8%. Sedangakan akurasi terendah sebesar 47,2% dangan perbandingan data latih dengan data uji 20%:80%.



Gambar 6.3 Grafik Hasil Pengujian Perbandingan Rasio Data

Pada grafik di Gambar 6.3 merupakan perbandingan hasil uji terhadap perbandingan rasio data latih dengan data uji terhadap akurasi implementasi ELM. Dapat ditarik kesimpulan bahwa perbandingan rasio data latih dengan data uji sangat berpengaruh karena setiap perubahan rasio data latih dengan data uji memiliki hasil akurasi yang berbeda- beda.

6.2.4 Pengujian Pengaruh Fungsi Aktivasi Terhadap Akurasi

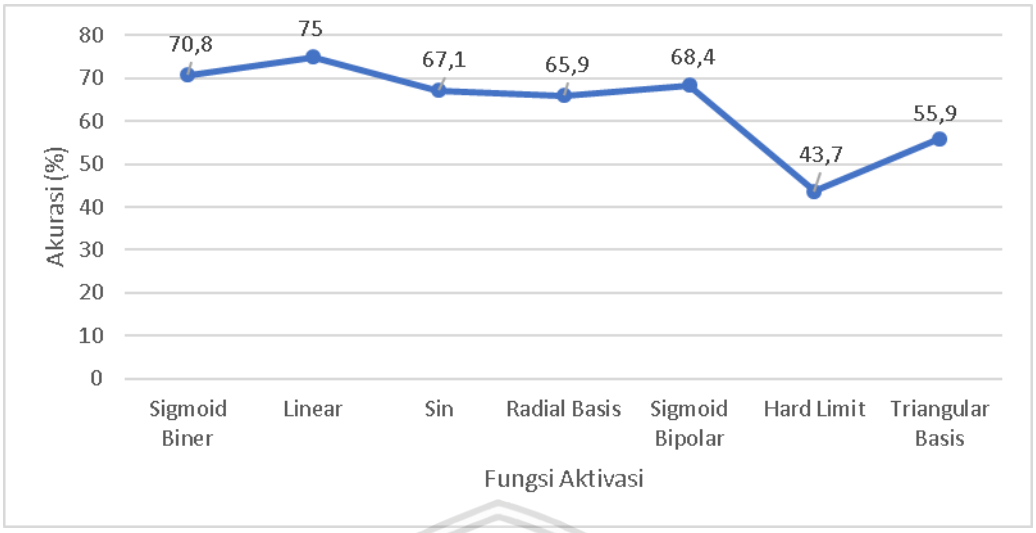
Pada pengujian ini memiliki maksud untuk mengetahui pengaruh fungsi aktivaei terhadap akurasi dalam implemantasi ELM. Pada algoritme ELM fungsi aktivasi digunakan pada tahapan menghitung nilai keluaran *hidden neuron* yaitu matriks H. Terdapat 7 fungsi aktivasi yang diuji pada pengujian ini. Ketujuh fungsi aktivasi yang diujikan adalah fungsi Sigmoid Biner, Linear, Sin, Radial Basis, Sigmoid Bipolar, Hard Limit, dan Triangular Basis. Perbandigan data latih dan data uji yang digunakan adalah rasio 80%:20% yang memiliki nilai akurasi tertinggi yang berdasarkan pengujian sebelumnya. Selain itu pada pengujian ini menggunakan *hidden neuron* sebanyak 10 yang mana juga memiliki nilai akurasi berdasarkan pengujian *hidden neuron.*

Pengujian pengaruh fungsi aktivasi terhadap nilai tingkat akurasi akan dilakukan sebanyak 10 kali percobaan yang nantinya akan diambil nilai rataan terbaik dari setiap fungsi aktivasi yang diujikan. Berikut ini merupakan hasil pengujian pengaruh fungsi aktivasi terharhadap nilai akurasi yang dapat dilihat pada Tabel 6.4.

Tabel 6.4 Pengaruh Fungsi Aktivasi Terhadap Akurasi

Percobaan ke-*i*	Nilai Akurasi Pada Fungsi Aktivasi						
	Sigmoid Biner	*Linear*	*Sin*	*Radial Basis*	*Sigmoid Bipolar*	*Hard Limit*	*Triangular Basis*
1	62	75	67	67	71	38	67
2	62	75	62	62	71	38	46
3	71	75	67	71	62	38	67
4	71	75	75	83	67	38	54
5	75	75	75	71	71	38	58
6	71	75	58	67	67	38	38
7	79	75	71	67	71	38	71
8	71	75	75	46	71	62	38
9	67	75	75	58	62	71	58
10	79	75	46	67	71	38	62
Akurasi	70,8	75	67,1	65,9	68,4	43,7	55,9

Dari hasil pengujian yang tertera pada Tabel 6.4 dapat dilihat bahwa fungsi aktivasi Linear memiliki rataan nilai akurasi tertinggi sebesar 75% dan fungsi aktivasi Hard Limit memiliki rataan nilai akurasi terendah sebesar 43,7%.



Gambar 6.4 Grafik Hasil Pengujian Pengaruh Fungsi Aktivasi Terhadap Akurasi

Pada grafik di Gambar 6.4 merupakan hasil pengujian fungsi aktivasi terhadap akurasi sistem Dapat ditarik kesimpulan bahwa secara garis besar perubahan fungsi aktivasi berpengaruh terhadap hasil akurasi. Hal ini disebabkan karena setiap fungsi aktivasi memiliki fungsi yang berbeda-beda.

BAB 7 KESIMPULAN DAN SARAN

Pada bab kesimpulan akan disampaikan hasil kesimpulan dan saran berdasarkan penelitian yang telah dilakukan mengenai implementasi algoritme *Extreme Learning Machine* untuk klasifikasi penanganan *Human Papilloma Virus*.

7.1 Kesimpulan

Kesimpulan yang dapat diambil pada penelitian ini setelah dilakukannya perancangan, implementasi dan analisis hasil pengujian terhadap implementasi algoritme ELM untuk klasifikasi penanganan *Human Papilloma Virus* (HPV) adalah sebagai berikut :

1. Pada permasalahan klasifikasi penangana *Human Papilloma Virus*, algoritme *Extreme Learning Machine* (ELM) dapat diimplementasikan. Proses klasifikasi metode penangana *Human Papilloma Virus*(HPV) dilakukan dengan cara menyipakan data latih dan data uji yang nantinya figunakan pada proses latihdan proses uji. Tahapan implemen tasi yang dilakuakn adalah sebagai berukut:
    a. Tahapan normalisasi terhadapa data yang digunakan.
    b. Tahapan latih
    c. Tahapan uji
    d. Tahapan perhitungan akurasi
2. Pada implementasi klasifikasi pmetode penangana *Human Papilloma Virus* (HPV) menggunakan ELM nantinya akan didapat nilai akurasi. Nilai akurasi yang didapat dari tahapan uji adalah sebagai berikut:
    a. Berdasarkan pengujian dan analisa perbandingan jumlah data latih dan data uji berpengaruh terhadap hasil perhitungan ELM. Hal tersebut terbukti dengan rasio data latih dan uji sebesar 80% berbanding 20% menghasilkan nilai akurasi terbaik sebesar 70,8% .
    b. Berdasarkan pengujian dan analisa jumlah hidden neuron berpengaruh terhadap hasil perhitungan ELM. Hal tersebut terbukti dengan jumlah hidden neuron 10 menghasilkan akurasi terbaik sebesar 69,1% dengan waktu proses selama 0,0390 detik, namun untuk *learning rate* terbaik didapat pada pengujian menggunakan 1 *hidden neuron* dengan *learning rate* selama 0,017 detik.
    c. Berdasarkan pengujian dan analisa fungsi aktivasi berpengaruh terhadap hasil perhitungan ELM. Hal tersebut terbukti dengan fungsi aktivasi linear didapatkan nilai akurasi terbaik dengan akurasi sebesar 75%.

7.2 Saran

Penelitian mengenai implementasi algoritme *Extreme Learning Machine* untuk klasifikasi dapat dikembangkan dengan beberapa saran sebagai berikut:

1. Menggunakan lebih data mengenai *Human Papilloma Virus* (HPV).
2. Melakukan perbandingan terhadap algoritme jaringan syaraf tiruan lainnya agar ditemukan perbandingan terbaik terhadap algoritme ELM baik terhadap akurasi maupun lama wktu proses
3. Melakukan penambahan algoritme optimasi untuk seleksi fitur atau optimasi parameter ELM berupa bobot masukan maupun bias agas hasil klasifikasi terhadap metode penanganan HPV menjadi lebih akurat maupun waktu proses yang menjadi lebih cepat.

DAFTAR PUSTAKA

Azad, F. J., Mahboubi, Y. & Khozeimeh, F., 2017. Intralesional immunotherapy compared to cryotherapy in the treatment of warts. *International Journal of Dermatology.*

Bourke, J. F., Berth-Jones, J. & Hutchinson, P. E., 1995. Cryotherapy of Common Viral Warts at Intervals of 1,2, and 3 Weeks. *British Journal of Dermatology.*

Christy, M., Adikara, P. P. & SetyaPerdana, R., 2017. Klasifikasi Penyimpangan Tumbuh Kembang Anak Menggunakan Metode Extreme Learning Machine (ELM).

Clifty, M., 2003. Immunotherapy for Recalcitrant Warts in Children Using Intralesional Mumps or Candida Antigens. *Pedriatic Dermatology.*

Huang, G.-B., Zhu, Q.-Y. & Siew, C.-K., 2004. Extreme Learning Machine: A New Learning scheme of Feedforward Neural Networks.

Khozeimeh, F. et al., 2017. An expert system for selecting wart treatment method.

Khozeimeh, F. et al., 2017. An Expert System for Selecting Wart Treatment Method.

Moore, S. W., 1998. *Griffith's Instructions For Patients.* Tucson: s.n.

Nur, A. F. S., Rizal, A. & Budiman, G., 2011. Klasifikasi Aritmia pada Sinyal EKG Menggunakan Ectreme Learning Machine.

Patro, G. K. & Kishore, K. S., 2015. Normalization : A Preprocessing Stage.

Prakoso, E., Wiesty, U. N. & Jondri, 2016. Klasifikasi Keadaan Mata Berdasarkan sinyal EEG menggunakan Extreme Learning Machines.

Thappa, D. M. & Chiramel, M. J., 2016. Evolving role of immunotherapy in the treatment of refractory warts.

Zhu, Q.-Y., Suganthan, P. & Huang, G.-B., 2005. Evolutionary Extreme Learning Machine.