IMPLEMENTASI JARINGAN SYARAF TIRUAN DENGAN METODE PEMBELAJARAN RPROP *(RESILIENT BACKPROPAGATION)* UNTUK KLASIFIKASI STATUS TAHAPAN KELUARGA SEJAHTERA

SKRIPSI

Untuk memenuhi sebagian persyaratan
untuk mencapai gelar Sarjana Komputer

Disusun oleh :

ALFA RIDHOANA
NIM.105090613111003

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
MALANG
2014

LEMBAR PERSETUJUAN

IMPLEMENTASI JARINGAN SYARAF TIRUAN DENGAN METODE PEMBELAJARAN RPROP *(RESILIENT BACKPROPAGATION)* UNTUK KLASIFIKASI STATUS TAHAPAN KELUARGA SEJAHTERA

SKRIPSI

Disusun oleh:

Alfa Ridhoana

NIM. 105090613111003

Skripsi ini telah disetujui oleh dosen pembimbing
pada tanggal 29 September 2014:

Dosen Pembimbing I,	Dosen Pembimbing II,
Rekyan Regasari M. P., S.T., M.T.	**Novanto Yudistira, S.Kom., M.Sc.**
NIK. 77041406120253	**NIK. 83111016110425**

LEMBAR PENGESAHAN

IMPLEMENTASI JARINGAN SYARAF TIRUAN DENGAN METODE PEMBELAJARAN RPROP *(RESILIENT BACKPROPAGATION)* UNTUK KLASIFIKASI STATUS TAHAPAN KELUARGA SEJAHTERA

SKRIPSI

Disusun oleh :

ALFA RIDHOANA

NIM. 105090613111003

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 16 Oktober 2014

Dosen Penguji I	Dosen Penguji II
Indriati, S.T., M.Kom.	**Denny Sagita R., S.Kom., M.Kom.**
NIK. 831013 06 1 2 0035	**NIK. 851124 06 1 1 0250**

Dosen Penguji III

Muhammad Tanzil Furqon, S.Kom., M.CompSc.
NIP. 19820930 200801 1 004

Mengetahui
Ketua Program Studi Teknik Informatika/Ilmu komputer

Drs. Mardji, M.T.
NIP. 19670801 199203 1 001

LEMBAR PERNYATAAN
ORISINALITAS SKRIPSI

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 dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan 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, 26 September 2014
Mahasiswa,

Alfa Ridhoana
NIM. 0710963041

KATA PENGANTAR

Alhamdulillahi robbil 'alamin. Puji syukur penulis panjatkan kehadirat Allah SWT, karena atas segala rahmat dan limpahan hidayahNya, penulis bisa menyelesaikan skripsi yang berjudul "Implementasi Jaringan Syaraf Tiruan dengan Metode Pembelajaran RPROP *(Resilient Backpropagation)* untuk Klasifikasi Status Tahapan Keluarga Sejahtera".

Skripsi ini disusun dan diajukan sebagai syarat untuk memperoleh gelar sarjana pada program studi Teknik Informatika/Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang.

Dalam penyelesaian skripsi ini, penulis telah mendapat begitu banyak bantuan baik moral maupun materiil dari banyak pihak. Atas bantuan yang telah diberikan, penulis ingin menyampaikan penghargaan dan ucapan terima kasih kepada:

1. Rekyan Regasari M. P., S.T., M.T. selaku pembimbing I dan Novanto Yudistira, S.Kom., M.Sc. selaku pembimbing II. Terima kasih atas semua waktu dan bimbingan yang telah diberikan dalam penyusunan skripsi ini.
2. Drs. Marji, M.T. selaku Ketua Program Studi Teknik Informatika/Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer.
3. Segenap bapak dan ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penulis.
4. Eyang dan segenap keluarga penulis. Terima kasih atas kasih sayang, doa, dan dukungan baik moril maupun materiil kepada penulis.
5. Segenap staf dan karyawan Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.
6. Rekan-rekan penulis, yaitu Eva F. Bisono, Heny Herawati, Jesicha Dwi Ayu Mayasari, dan Anggreyni yang telah memberikan dukungannya secara moril dan memahami suka duka penulis selama mengerjakan skripsi ini.
7. Rekan-rekan Teknik Informatika/Ilmu Komputer 2010 Universitas Brawijaya yang telah memberikan dukungannya dalam penyusunan skripsi ini.

8. Pihak lain yang telah membantu terselesaikannnya skripsi ini yang tidak bisa penulis sebutkan satu-persatu.

Penulis menyadari bahwa masih banyak kekurangan dalam laporan ini disebabkan oleh keterbatasan kemampuan dan pengalaman. Oleh karena itu, saran dan kritik yang sifatnya membangun demi penulisan dan mutu skripsi ini untuk kelanjutan penelitian serupa di masa mendatang sangat diharapkan. Penulis berharap semoga skripsi ini dapat memberikan manfaat kepada pembaca, baik oleh penulis sendiri maupun pihak-pihak lainnya.

Malang, 26 September 2014

Penulis

IMPLEMENTASI JARINGAN SYARAF TIRUAN DENGAN METODE PEMBELAJARAN RPROP *(RESILIENT BACKPROPAGATION)* UNTUK KLASIFIKASI STATUS TAHAPAN KELUARGA SEJAHTERA

Alfa Ridhoana. 2014. Implementasi Jaringan Syaraf Tiruan dengan Metode Pembelajaran RPROP *(Resilient Backpropagation)* untuk Status Tahapan Keluarga Sejahtera. Skripsi Program Studi Teknik Informatika/Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
Pembimbing: Rekyan Regasari M. P., S.T., M.T. dan
Novanto Yudistira, S.Kom., M.Sc.

ABSTRAK

Tingginya angka kelahiran menyebabkan permasalahan pembangunan kependudukan di Indonesia. Hal ini berakibat pada beratnya beban pemerintah untuk menurunkan tingkat kemiskinan dan meningkatkan kesejahteraan masyarakat. Salah satu upaya pemerintah untuk mewujudkan program pembangunan kependudukan di Indonesia adalah melaksanakan pendataan keluarga setiap tahun di seluruh wilayah Indonesia. Dengan demikian hasil pendataan keluarga akan berguna bagi keluarga, masyarakat, dan pemerintah untuk meningkatkan kualitas keluarga. Untuk proses pengklasifikasiannya, petugas harus mempertimbangkan 21 nilai indikator yang telah diperoleh dari masing-masing keluarga. Pada penelitian ini mengimplementasikan sebuah sistem untuk mengklasifikasikan status tahapan keluarga sejahtera dengan algoritma JST *Resilient Backpropagation*. Sistem ini dapat memprediksi dengan baik status tahapan keluarga sejahtera dengan menggunakan data latih *balanced class* sebesar 70% dan data uji sebesar 30%, sehingga hasil prediksi dapat mencapai tingkat rata-rata akurasi prediksi dari 5 kali percobaan sebesar 90.33% dan akurasi tertinggi dari 5 kali percobaan sebesar 93.33%. Konfigurasi jaringan yang dihasilkan dalam penelitian ini, yaitu 21 unit neuron pada *input layer*, 45 unit neuron pada *hidden layer*, dan 1 unit neuron pada *output layer* dengan *learning rate* sebesar 0.8.

Kata Kunci: Keluarga Sejahtera, Jaringan Syaraf Tiruan, *Resilient Backpropagation*

IMPLEMENTATION OF ARTIFICIAL NEURAL NETWORK USING RPROP (RESILIENT BACKPROPAGATION) LEARNING METHOD FOR STATUS CLASSIFICATION OF FAMILY WELFARE STAGES

Alfa Ridhoana. 2014. Implementation of Artificial Neural Network Using Rprop (Resilient Backpropagation) Learning Method for Status Classification of Family Welfare Stages. Skripsi Study Program of Information Techniques/Computer Science. Information Technology and Computer Science Brawijaya University.
Supervisor: Rekyan Regasari M. P., S.T., M.T. and
Novanto Yudistira, S.Kom., M.Sc.

ABSTRACT

The high rate of birth led to the development of population problems in Indonesia. It makes the government have a serious responsibility to reduce the poverty and improves the social welfare. One of the government's efforts to realize the population development program in Indonesia is collecting the data of all families throughout Indonesia each year. Thus, the results of the data collection will be useful to every family, community, and government to improve the quality of the family. In the classification process, the officer should consider 21 indicators which was obtained of each family. This research has implemented a system to classify the stages of family welfare with Resilient Backpropagation neural network algorithm. This system is able to predict the stages of family welfare status well using 70% of data training with *balanced class* and 30% of data testing, so that the result of these predictions achieve 90.33% for the average accuracy of 5 times experiment and 93.33% for the highest accuracy. The network structures that has been resulting in this research are 21 units of neurons in input layer, 45 units of neurons in hidden layer, and 1 unit of neuron in output layer with learning rate of 0.8.

Keywords: Family Welfare, Neural Network, Resilient Backpropagation

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SOURCE CODE

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Tingginya angka kelahiran di Indonesia menyebabkan permasalahan pembangunan kependudukan di Indonesia. Hal ini berakibat pada beratnya beban pemerintah dalam menurunkan tingkat kemiskinan dan meningkatkan kesejahteraan masyarakat. Program pembangunan keluarga sejahtera semakin mendapat pijakan yang kuat dengan diundangkannya UU No. 10 tahun 1992 tentang perkembangan kependudukan dan pembangunan keluarga sejahtera [SUN-07]. Salah satu upaya pemerintah untuk mewujudkan program pembangunan kependudukan di Indonesia adalah dengan melaksanakan pendataan keluarga setiap tahunnya di seluruh wilayah Indonesia. Pendataan keluarga dilakukan secara langsung dengan mendatangi keluarga-keluarga melalui kunjungan dari rumah ke rumah oleh para kader atau petugas pendata setempat. Data untuk aspek keluarga sejahtera dikumpulkan dengan menggunakan 21 indikator sesuai pemikiran para pakar sosiolog [SUB-10].

Dengan adanya pengklasifikasian keluarga berdasarkan status kesejahteraannya, yaitu tahap keluarga prasejahtera, keluarga sejahtera I, keluarga sejahtera II, keluarga sejahtera III, dan keluarga sejahtera III plus, maka dapat memberikan informasi mengenai laporan hasil pengklasifikasian atau pengelompokan penduduk, sehingga pemerintah dapat meningkatkan kualitas pelaksanaan program pembangunan keluarga sejahtera. Selain itu, hasil pengklasifikasian keluarga akan berguna bagi keluarga dan pemerintah untuk membantu dirinya dalam menuntaskan keluarga dari ketertinggalan dan meningkatkan kualitas keluarga berdasarkan data yang telah terkelompokkan.

Pengklasifikasian keluarga yang dilakukan oleh para kader atau petugas setempat (Posyandu kecamatan Bumiaji, Kota Batu) dirasa masih menyulitkan petugas. Petugas harus mempertimbangkan 21 nilai indikator yang telah diperoleh dari masing-masing keluarga. Jumlah keluarga dalam suatu wilayah desa atau kelurahan tidaklah sedikit, tentunya hal ini akan memakan waktu yang cukup lama bila proses pengklasifikasian. Ketentuan dari instansi yang dijadikan acuan

untuk proses pengklasifikasian biasanya masih dianggap rancu atau terkadang belum sesuai menurut perasaan si petugas yang bersangkutan, sehingga menimbulkan interpretasi tersendiri untuk mengklasifikasikannya. Sebenarnya sudah ada aplikasi online khusus dari pemerintah yang digunakan untuk manajemen pemutakhiran data keluarga yang beralamat pada http://aplikasi.bkkbn.go.id/, namun aplikasi ini belum mendukung fitur pengklasifikasian otomatis.

Oleh karena itu, perlu dilakukan penelitian untuk menghasilkan suatu aplikasi yang bisa memudahkan dan menjamin akurasi pengklasifikasian keluarga berdasarkan status kesejahteraannya sesuai dengan pola yang ada. Jaringan Syaraf Tiruan adalah salah satu metode yang bisa diaplikasikan untuk menyelesaikan permasalahan klasifikasi.

Jaringan Syaraf Tiruan atau *Artificial Neural Network* merupakan suatu sistem pemrosesan informasi yang memiliki karakteristik menyerupai jaringan syaraf manusia. Jaringan syaraf tiruan terorganisasi layaknya anatomi otak manusia, sehingga jaringan syaraf tiruan mampu belajar dari pengalaman dan melakukan generalisasi berdasarkan contoh-contoh yang diperolehnya [SUB-08]. Dengan kemampuan tersebut, jaringan syaraf tiruan dapat melakukan regresi non-linear terhadap pola-pola parameter untuk menyelesaikan kasus klasifikasi status tahapan keluarga sejahtera.

Penelitian yang dilakukan oleh Effendy, dkk. (2008) untuk prediksi penyakit jantung koroner berdasarkan faktor risiko dengan jaringan syaraf tiruan *backpropagation* dapat mengenali 80% dari data uji sebanyak 20.

Adapun kelemahan dari metode *backpropagation* adalah penggunaan fungsi aktivasi sigmoid menyebabkan *gradient* akan mendekati nol ketika input yang diberikan sangat banyak, sehingga *gradient* yang bernilai nol akan menyebabkan rendahnya perubahan bobot [FEB-07]. Apabila bobot-bobot tidak cukup mengalami perubahan, maka algoritma akan sangat lambat untuk mendekati nilai konfigurasi jaringan optimumnya. Oleh karena itu, diperlukan suatu metode pembelajaran yang mampu mengatasi kelemahan jaringan syaraf tiruan *backpropagation*, yaitu dengan menggunakan metode pembelajaran Rprop (*resilient backpropagation*).

Metode pembelajaran Rprop (*resilient backpropagation)* adalah algoritma pembelajaran yang bertujuan untuk menghilangkan pengaruh *gradient* terhadap perubahan bobot dengan hanya menggunakan tanda turunannya saja. Tanda turunan itu yang akan menentukan arah dari perubahan bobot [FEB-07].

Penelitian yang dilakukan oleh Virgiandhana (2013) untuk diagnosis penyakit diabetes mellitus dengan jaringan syaraf tiruan *resilient backpropagation* mampu menghasilkan tingkat akurasi prediksi sebesar 96.66% dari data uji sebanyak 30 data.

Oleh karena itu, implementasi klasifikasi status tahapan keluarga sejahtera menggunakan metode pembelajaran *resilient backpropagation* dimaksudkan untuk mendapatkan konfigurasi jaringan optimum dengan pembelajaran yang ditentukan oleh tanda *gradient*. Konfigurasi jaringan optimal yang dimaksud pada penelitian ini adalah kombinasi jumlah *hidden neuron* dan nilai *learning rate* yang akan diperoleh dengan melakukan metode *trial and error*.

Berdasarkan latar belakang yang telah diuraikan di atas, maka judul yang diambil untuk skripsi ini adalah **"Implementasi Jaringan Syaraf Tiruan dengan Metode Pembelajaran RPROP (*Resilient Backpropagation*) untuk Klasifikasi Status Tahapan Keluarga Sejahtera"**.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, maka rumusan masalah pada skripsi ini adalah :

1. Bagaimana mengimplementasikan algoritma pembelajaran *Resilient Backpropagation* untuk klasifikasi status tahapan keluarga sejahtera.
2. Bagaimana tingkat akurasi algoritma pembelajaran *Resilient Backpropagation* untuk klasifikasi status tahapan keluarga sejahtera.

1.3 Batasan Masalah

Batasan masalah pada skripsi ini adalah :

1. Parameter atau indikator yang digunakan untuk pengklasifikasian status tahapan keluarga sejahtera adalah 21 indikator yang sudah ditetapkan oleh

BKKBN (Badan Kependudukan dan Keluarga Berencana Nasional) pada format formulir F/I/MDK/2011.

2. Tidak menangani data yang memiliki *missing value* pada data latih dan data uji.
3. Data yang digunakan bersumber dari data hasil pemutakhiran keluarga tahun 2013 di Desa/Kelurahan Tulungrejo, RT. 01, 02, 03, 04, 05 - RW. 05, Kecamatan Bumi Aji, Kota Batu.
4. Hasil akhir dari klasifikasi status tahapan keluarga sejahtera ini tidak dibandingkan dengan metode lainnya.
5. Fungsi aktivasi yang digunakan adalah fungsi aktivasi sigmoid biner.

1.4 Tujuan

Tujuan yang ingin dicapai dari penulisan skripsi ini adalah :

1. Menerapkan algoritma pembelajaran *Resilient Backpropagation* untuk klasifikasi status tahapan keluarga sejahtera berdasarkan input parameter yang diberikan.
2. Mengetahui tingkat akurasi algoritma pembelajaran *Resilient Backpropagation* untuk klasifikasi status tahapan keluarga sejahtera.

1.5 Manfaat

Manfaat yang diharapkan untuk bisa diambil dari penelitian ini adalah menghasilkan sebuah software yang menerapkan algoritma pembelajaran JST *Resilient Backpropagation* untuk menyelesaikan masalah pengklasifikasian status tahapan keluarga sejahtera, sehingga diharapkan dapat membantu kinerja petugas pendataan keluarga sejahtera untuk menghasilkan pengklasifikasian keluarga yang akurat dan tepat sesuai pola.

1.6 Metodologi Pemecahan Masalah

Untuk mencapai tujuan yang dirumuskan sebelumnya, maka metodologi yang digunakan dalam penulisan skripsi ini adalah :

1. Identifikasi Permasalahan

   Mengenali permasalahan tentang status tahapan keluarga sejahtera untuk menemukan ruang lingkup masalah, kemudian dipilih salah satu masalah yang sesuai dengan kemampuan peneliti.

2. Studi Literatur

   Mempelajari klasifikasi status tahapan keluarga sejahtera beserta indikator-indikator yang mempengaruhinya dan mempelajari teori jaringan syaraf tiruan *Resilient Backpropagation* dari berbagai referensi.

3. Analisa dan Perancangan Sistem

   Membuat perancangan sistem dengan analisisnya yang terstruktur.

4. Implementasi

   Mengimplementasikan hasil rancangan sistem dalam suatu program komputer.

5. Uji Coba dan Analisa Hasil Implementasi

   Menguji perangkat lunak pada beberapa data yang berbeda dan menganalisa hasil dari implementasi tersebut kemudian melakukan evaluasi.

1.7 Sistematika Penulisan

Skripsi ini disusun berdasarkan sistematika penulisan sebagai berikut :

1. **BAB I PENDAHULUAN**

   Berisi latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi pemecahan masalah, dan sistematika penulisan.

2. **BAB II KAJIAN PUSTAKA DAN DASAR TEORI**

   Pada bagian kajian pustakan menguraikan mengenai penelitian-penelitian sebelumnya yang terkait, sehingga bisa dijadikan acuan dalam penelitian. Pada dasar bagian teori menguraikan teori-teori yang berhubungan dengan tahapan keluarga sejahtera dan algoritma pembelajaran *Resilient Backpropagation*.

3. **BAB III METODE PENELITIAN DAN PERANCANGAN**

   Pada bab ini dijelaskan mengenai metode-metode dan perancangan yang digunakan dalam mengimplementasikan algoritma pembelajaran *Resilient Backpropagation* untuk klasifikasi status tahapan keluarga sejahtera.

4. **BAB IV IMPLEMENTASI**

   Pada bab ini dijelaskan mengenai implementasi program untuk klasifikasi status tahapan keluarga sejahtera,

5. **BAB V PENGUJIAN DAN ANALISIS**

   Pada bab ini dijelaskan mengenai pengujian dan analisa hasil dari pengujian sistem perangkat lunak yang dibangun.

5. **BAB V PENUTUP**

   Bab ini berisi kesimpulan dari seluruh rangkaian penelitian serta saran yang memungkinkan untuk pengembangan penelitian di masa mendatang.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 Kajian Pustaka

Jaringan syaraf tiruan tersusun atas elemen-elemen sederhana yang beroperasi secara paralel dengan menirukan cara kerja otak manusia. JST dapat digunakan untuk menyelesaikan masalah-masalah kompleks di berbagai bidang, antara lain identifikasi, pengenalan pola, klasifikasi, restorasi citra, termasuk juga untuk sistem kendali [SUP-12].

Pada penelitian sebelumnya yang dilakukan oleh Effendy, dkk. (2008) untuk prediksi penyakit jantung koroner berdasarkan faktor risiko dengan jaringan syaraf tiruan *backpropagation*. Jaringan syaraf tiruan dilatih agar mengenali pola 9 faktor risiko penderita penyakit jantung koroner atau 9 pola faktor risiko orang sehat. Pelatihan dilakukan dengan beberapa tahapan, antara lain dengan identifikasi bobot dan bias dengan menggunakan fungsi *newff*, kemudian dilatihkan dengan fungsi *trainlm* dengan menentukan masukan dan target berupa matriks. Hasil pelatihan digunakan untuk mencari konfigurasi terbaik dengan cara mengubah konstanta belajar serta lapisan tersembunyi secara *trial* dan *error*. Hasil pengujian menunjukkan bahwa 4 data (20%) tidak sesuai dengan target dan 16 data (80%) sesuai dengan target. Hal ini sebenarnya disebabkan oleh beberapa faktor, antara lain karena variabel input yang berjumlah 9 unit sel, maka idealnya data yang dilatihkan tidak hanya 40 data rekam medis pasien. Permasalahannya adalah semakin banyak data yang dilatihkan, maka membutuhkan waktu pelatihan yang cukup lama pada metode *backpropagation* ini.

Adapun kelemahan dari metode *backpropagation* adalah penggunaan fungsi aktivasi sigmoid menyebabkan *gradient* akan mendekati nol ketika input yang diberikan sangat banyak, sehingga *gradient* yang bernilai nol akan menyebabkan rendahnya perubahan bobot [FEB-07]. Apabila bobot-bobot tidak cukup mengalami perubahan, maka algoritma akan sangat lambat untuk mendekati konfigurasi jaringan optimumnya. Oleh karena itu, diperlukan suatu metode pembelajaran yang mampu mengatasi kelemahan jaringan syaraf tiruan *backpropagation*, yaitu dengan menggunakan metode pembelajaran Rprop

(*resilient backpropagation*). Metode pembelajaran Rprop (*resilient backpropagation)* adalah algoritma pembelajaran yang bertujuan untuk menghilangkan pengaruh *gradient* terhadap perubahan bobot [FAD-09].

Pada penelitian sebelumnya yang dilakukan oleh Virgiandhana (2013) untuk diagnosis penyakit *diabetes mellitus* dengan jaringan syaraf tiruan *resilient backpropagation.* Metode jaringan syaraf tiruan (JST) dengan algoritma *resilient backpropagation* dapat diimplementasikan untuk mendiagnosa kasus diabetes dengan model jaringan yang digunakan yaitu 4 unit neuron pada *input layer,* 70 unit neuron pada *hidden layer,* dan 1 unit neuron pada *output layer*. Kombinasi parameter yang terbaik hasil implementasi metode ini yaitu dengan nilai *learning rate* sebesar 0.9 dan nilai neuron pada *hidden layer* sebanyak 70 unit. Dengan data latih sebesar 40 sampai 70 dan data uji sebesar 30 diperoleh nilai akurasi sebesar 96.66%. Hasil pengenalan sistem dalam memprediksi diabetes yang terbesar adalah sebesar 96.66% didapat dari jumlah prediksi yang benar adalah 29 dan jumlah prediksi yang salah adalah 1. Dengan tingkat akurasi yang tinggi, yaitu 96.66%, menunjukkan bahwa metode *resilient backpropagation* sangat baik untuk diimplementasikan pada kasus diagnosis penyakit *diabetes mellitus* ini.

2.2 Dasar Teori

2.2.1 Keluarga Sejahtera

2.2.1.1 Definisi Kesejahteraan dan Keluarga Sejahtera

Terdapat beberapa pengertian mengenai kesejahteraan, karena kesejahteraan lebih bersifat subyektif. Tujuan dan cara hidup masyarakat yang berbeda-beda akan memberikan nilai-nilai yang berbeda pula tentang kesejahteraan. Kesejahteraan merupakan sejumlah kepuasan yang diperoleh seseorang dari hasil mengkonsumsi pendapatan yang diterima, sehingga kesejahteraan bersifat relatif, karena tergantung dari besarnya kepuasan yang diperoleh dari hasil mengkonsumsi pendapatan tersebut [SUN-07]. Kesejahteraan adalah suatu tata kehidupan dan penghidupan sosial, material, maupun spiritual yang diliputi rasa keselamatan, kesusilaan dan ketentraman lahir batin yang memungkinkan setiap warga negara untuk mengadakan usaha-usaha pemenuhan

kebutuhan jasmani, rohani dan sosial yang sebaik-baiknya bagi diri, rumah tangga serta masyarakat [SUN-07].

Pengertian keluarga sejahtera menurut BKKBN (Badan Kependudukan dan Keluarga Berencana Nasional) adalah keluarga yang dapat memenuhi kebutuhan anggotanya baik kebutuhan sandang, pangan, perumahan, sosial dan agama, keluarga yang penghasilan dengan jumlah anggota keluarganya berkeseimbangan, keluarga yang dapat memenuhi kebutuhan kesehatan anggota keluarganya, kehidupan sosial bersama dengan masyarakat sekitar, melakukan ibadah sesuai agamanya di samping kebutuhan pokok.

2.2.1.2 Macam-macam Status Tahapan Keluarga Sejahtera

BKKBN (Badan Kependudukan dan Keluarga Berencana Nasional) merumuskan konsep keluarga sejahtera dengan mengelompokkan status keluarga ke dalam lima tahapan, yaitu keluarga prasejahtera (KPS), keluarga sejahtera I (KS-I), keluarga sejahtera II (KS-II), keluarga sejahtera III (KS-III), dan keluarga sejahtera III plus (KS-III Plus). Aspek keluarga sejahtera dikumpulkan dengan menggunakan 21 indikator sesuai denga pemikiran para sosiolog dalam membangun keluarga sejahtera dengan mengetahui faktor-faktor dominan yang menjadi kebutuhan setiap keluarga [SUB-10].

2.2.1.3 Indikator Tahapan Keluarga Sejahtera

Indikator tahapan keluarga sejahtera menurut BKKBN adalah sebagai berikut (berjumlah 21 indikator) :

a. Tahapan Keluarga Pra Sejahtera

   Keluarga yang belum dapat memenuhi salah satu indikator tahapan Keluarga Sejahtera I.

b. Tahapan Keluarga Sejahtera I

   Keluarga yang baru dapat memenuhi indikator-indikator berikut :

   1. Pada umumnya anggota keluarga makan dua kali sehari atau lebih.
   2. Anggota keluarga memiliki pakaian yang berbeda untuk di rumah, bekerja/sekolah dan bepergian.

3. Rumah yang ditempati keluarga mempunyai atap, lantai dan dinding yang baik.
4. Bila ada anggota keluarga sakit dibawa ke sarana kesehatan.
5. Bila pasangan usia subur ber-KB pergi ke sarana pelayanan kontrasepsi.
6. Semua anak umur 7-15 tahun dalam keluarga bersekolah.

c. Tahapan Keluarga Sejahtera II

Keluarga yang sudah dapat memenuhi indikator tahapan Keluarga Sejahtera I (indikator 1 sampai dengan 6) dan indikator berikut :

7. Pada umumnya anggota keluarga melaksanakan ibadah sesuai dengan agama dan kepercayaannya.
8. Paling kurang sekali seminggu seluruh keluarga makan daging ikan/telur.
9. Seluruh anggota keluarga memperoleh paling kurang satu stel pakaian baru dalam setahun.
10. Luas lanti rumah paling kurang 8 m^2 setiap penghuni rumah.
11. Tiga bulan terakhir keluarga dalam keadaan sehat sehingga dapat melaksanakan tugas/ fungsinya.
12. Ada seorang atau lebih anggota keluarga yang bekerja untuk memperoleh penghasilan.
13. Seluruh anggota keluarga umur 10-60 tahun bisa baca tulis latin.
14. Pasangan usia subur dengan anak 2 atau lebih menggunakan alat/ obat kontrasepsi.

d. Tahapan Keluarga Sejahtera III

Keluarga yang sudah memenuhi indikator tahapan Keluarga Sejahtera I dan indikator Keluarga Sejahtera II (indikator 1 sampai dengan 14) dan indikator berikut ini :

15. Keluarga berupaya meningkatkan pengetahuan agama.
16. Sebagian penghasilan keluarga ditabung dalam bentuk uang maupun barang.
17. Kebiasaan keluarga makan bersama paling kurang seminggu sekali dimanfaatkan untuk berkomunikasi.

18. Keluarga sering ikut dalam kegiatan masyarakat di lingkungan tempat tinggal.
19. Keluarga memperoleh informasi dari surat kabar/ majalah/ radio/ TV.

e. Tahapan Keluarga Sejahtera III Plus

Keluarga yang dapat memenuhi indikator tahapan Keluarga Sejahtera I, indikator Keluarga Sejahtera II dan indikator Keluarga Sejahtera III (indikator 1 sampai dengan 19) dan indikator berikut :

20. Keluarga secara teratur dengan sukarela memberikan sumbangan materi untuk kegiatan sosial.
21. Ada anggota keluarga yang aktif sebagai pengurus Perkumpulan Sosial/ Yayasan/ Institusi Masyarakat.

Aturan tata cara pengklasifikasian status tahapan keluarga sejahtera oleh BKKBN, yaitu jika tidak dapat memenuhi satu atau lebih dari 6 indikator KS-I (Keluarga Sejahtera I), maka termasuk ke dalam Keluarga Prasejahtera. Jika tidak dapat memenuhi satu atau lebih dari 8 indikator KS-II (Keluarga Sejahtera II), maka termasuk ke dalam KS-I (Keluarga Sejahtera I). Jika tidak dapat memenuhi satu atau lebih dari 5 indikator KS-III (Keluarga Sejahtera III), maka termasuk ke dalam KS-II (Keluarga Sejahtera II). Jika tidak dapat memenuhi satu atau lebih dari 2 indikator KS-III Plus (Keluarga Sejahtera III Plus), maka termasuk ke dalam KS-III (Keluarga Sejahtera III).

Ketentuan atau aturan tata cara pengklasifikasian status tahapan keluarga sejahtera dari BKKBN tersebut belum tentu bisa digunakan pada setiap daerah. Hal ini karena pada masing-masing instansi (Posyandu) biasanya memiliki pakar-pakar sendiri untuk memahami masing-masing kriteria pengklasifikasian berdasarkan 21 indikator yang ada, sehingga terkadang tata cara pengklasifikasian status tahapan keluarga sejahtera mengikuti aturan yang telah ditentukan oleh pakar.

2.2.2 Jaringan Syaraf Tiruan

2.2.2.1 Definisi Jaringan Syaraf Tiruan

Sejak ditemukan pertama kali oleh McCulloch dan Pitts pada tahun 1948, JST telah berkembang pesat dan telah digunakan pada banyak aplikasi. Jaringan syaraf tiruan (JST) telah dikembangkan sejak tahun 1940. Jaringan Syaraf Tiruan atau *Artificial Neural Network* merupakan suatu sistem pemrosesan informasi yang memiliki karakteristik seperti jaringan syaraf manusia. Jaringan syaraf tiruan mampu belajar dari pengalaman dan melakukan generalisasi berdasarkan contoh-contoh yang diperolehnya [SUB-08]. Pengalaman yang dimaksud dalam jaringan syaraf tiruan adalah berupa data masa lalu. Data masa lalu akan dipelajari oleh jaringan staraf tiruan, sehingga memiliki kemampuan untuk memberikan keputusan terhadap data yang belum pernah dipelajari.

Jaringan syaraf tiruan merupakan salah satu metode dalam pengklasifikasian objek yang digunakan sebagai model pengambilan keputusan. Klasifikasi objek bertujuan untuk mengelompokkan objek ke dalam kelompok yang sesuai berdasarkan ciri-ciri yang dimiliki objek tersebut [FEB-07].

Secara garis besar pada jaringan syaraf tiruan terdapat dua tahap pemrosesan informasi, yaitu [FEB-07] :

a. Tahap belajar

   Tahap ini dimulai dengan memasukkan pola-pola belajar ke dalam jaringan. Pola-pola yang sudah dimasukkan ini akan memberikan kemampuan pada jaringan untuk mengubah bobot yang menjadi penghubung antar node.

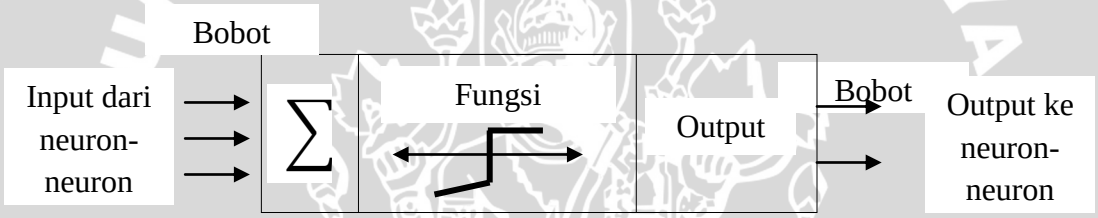
b. Tahap pengenalan

   Pada tahap ini dilakukan pengenalan terhadap suatu pola masukan dengan menggunakan bobot hasil tahap belajar.

2.2.2.2 Komponen Jaringan Syaraf Tiruan

Terdapat beberapa tipe jaringan syaraf tiruan, dari kesemuanya memiliki komponen-komponen yang hampir sama. Seperti halnya otak manusia, jaringan syaraf tiruan juga terdiri dari beberapa neuron dan terdapat hubungan antar neuron tersebut. Neuron-neuron tersebut akan mentransformasikan informasi yang diterima melalui sambungan menuju neuron-neuron yang lain, sehingga dalam

jaringan syaraf hubungan seperti ini dinamakan dengan bobot. Neuron-neuron buatan tersebut bekerja dengan cara yang sama pula dengan neuron biologis. Informasi akan dikirim ke neuron dengan bobot kedatangan tertentu. Informasi yang telah masuk akan diproses oleh suatu fungsi perambatan yang akan menjumlahkan nilai-nilai semua bobot yang datang. Hasil penjumlahan ini kemudian akan dibandingkan dengan suatu nilai ambang (*threshold*) tertentu melalui fungsi aktivasi setiap neuron. Apabila input tersebut melewati suatu nilai ambang tertentu, maka neuron tersebut akan diaktifkan, tapi kalau tidak, maka neuron tersebut tidak akan diaktifkan. Apabila neuron tersebut diaktifkan, maka neuron tersebut akan mengirimkan output melalui bobot-bobot outputnya ke semua neuron yang berhubungan dengannnya [PAN-08]. Gambar 2.1 menunjukkan struktur neuron pada jaringan syaraf tiruan.



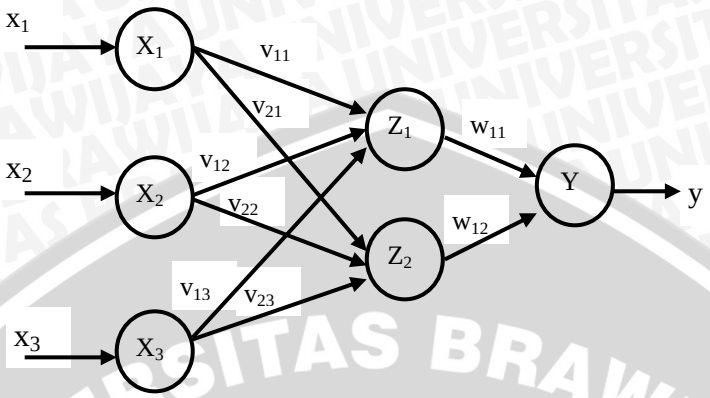
Gambar 2. 1 Struktur neuron jaringan syaraf tiruan
Sumber : [SIN-09]

Pada jaringan syaraf tiruan, neuron-neuron akan dikumpulkan dalam lapisan (*layer*) yang disebut dengan lapisan neuron (*neuron layer*). Neuron-neuron pada satu lapisan akan dihubungkan dengan lapisan-lapisan sebelum dan sesudahnya (kecuali lapisan input dan lapisan output). Informasi yang diberikan pada jaringan syaraf akan dirambatkan lapisan ke lapisan, mulai dari lapisan input sampai ke lapisan output melalui lapisan lainnya, yang sering disebut sebagai lapisan tersembunyi (*hidden layer*) [SIN-09].

2.2.2.3 Arsitektur Jaringan Syaraf Tiruan

Neuron-neuron dalam suatu sistem jaringan syaraf dikelompokkan dalam lapisan-lapisan. Faktor yang akan menentukan keadaan dari suatu neuron adalah fungsi aktivasi dan pola bobotnya. Setiap *neuron* pada satu lapisan harus dipetakan tepat ke setiap *neuron* pada lapisan berikutnya. Adapun arsitektur dari

jaringan syaraf tiruan dengan banyak lapisan (*multi layer network*) sebagai berikut [SUP-12] :



Gambar 2. 2 Contoh jaringan dengan banyak lapisan
Sumber : [SUP-12]

Pada jaringan dengan banyak lapisan memiliki 1 atau lebih lapisan (*hidden layer*) yang terletak di antara lapisan input dan lapisan output. Gambar 2.2 merupakan contoh jaringan dengan banyak lapisan.

Keterangan:

x_1, x_2, x_3	= nilai *input*
X_1, X_2, X_3	= lapisan *input*
$v_{11}, v_{12}, v_{13}, v_{21}, v_{22}, v_{23}$	= matriks bobot antara *input* dan *hidden layer*
Z_1, Z_2	= *hidden layer*
w_1, w_2	= matriks bobot antara *hidden layer* dan *output*
Y	= lapisan *output*
y	= nilai *output*

2.2.2.4 Metode Pembelajaran

Ciri utama dari sebuah sistem jaringan syaraf tiruan adalah kemampuan untuk belajar dari pengalaman terdahulu. Agar sistem dapat berfungsi dengan baik, jaringan dilatih dengan menggunakan data masa lalu (pembelajaran/pelatihan). Proses belajar dalam jaringan syaraf tiruan dikategorikan menjadi tiga jenis, yaitu [PUS-06] :

a. *Supervised learning* (pembelajaran terawasi)

Pada pembelajaran ini, setiap pola yang diberikan ke dalam JST telah diketahui outputnya (target yang telah diketahui sebelumnya). Selisih antara pola output aktual (output yang dihasilkan) dengan pola output target yang disebut *error* digunakan untuk mengoreksi bobot JST, sehingga JST mampu menghasilkan output sedekat mungkin dengan pola target yang telah diketahui oleh JST. Contoh algoritma JST yang menggunakan metode ini adalah : Hebbian, Perceptron, ADALINE, Boltzman, Hopfield, *Backpropagation*.

b. *Unsupervised learning* (pembelajaran tak terawasi)

Pada pembelajaran ini, jaringan tidak memerlukan target output, sehingga tidak dapat ditentukan hasil seperti apakah yang diharapkan selama proses pembelajaran. Selama proses pembelajaran, nilai bobot disusun dalam suatu range tertentu. Nilai bobot tersebut tergantung pada nilai input yang diberikan. Tujuan pembelajran ini adalah mengelompokkan unit-unit yang hampir sama dalam suatu area tertentu. Pembelajaran ini biasanya sangat cocok untuk klasifikasi pola. Contoh algoritma JST yang menggunakan metode ini adalah: *Competitive*, Hebbian, Kohonen, LVQ (*Learning Vector Quantization*), Neocognitron.

c. *Hybrid learning*

Metode pembelajaran ini merupakan kombinasi dari metode pembelajaran *supervised learning* dan *unsupervised learning*. Sebagian dari bobot-bobotnya ditentukan melalui *supervised learning* dan sebagian lainnya melalui *unsupervised learning*. Contoh algoritma JST yang menggunakan metode ini yaitu : algoritma RBF.

2.2.2.5 Fungsi Aktivasi

Fungsi aktivasi merupakan fungsi matematis yang berfungsi untuk membatasi dan menentukan jangkauan keluaran suatu neuron. Fungsi aktivasi yang sering digunakan dalam jaringan syaraf tiruan adalah fungsi sigmoid biner dan fungsi sigmoid bipolar yang ditunjukkan pada Persamaan (2-1) sampai Persamaan (2-4) [SIA-05] :

a. Fungsi Sigmoid Biner

Fungsi aktivasi ini memiliki range (0, 1).

$$f(x) = \frac{1}{1+e^{(-x)}} \quad (2\text{-}1)$$

dengan turunannya

$$f'(x) = f(x)[1-f(x)] \quad (2\text{-}2)$$

b. Fungsi Sigmoid Bipolar

Fungsi aktivasi ini memiliki range (-1, 1).

$$f(x) = \frac{1-e^{(-x)}}{1+e^{(-x)}} \quad (2\text{-}3)$$

dengan turunannya

$$f'(x) = \frac{1}{2}[1+f(x)][1-f(x)] \quad (2\text{-}4)$$

Keterangan :

e = bilangan eural yaitu 2.71828

x = hasil penjumlahan dari sinyal-sinyal input terbobot.

f(x) = fungsi untuk mengaktivasi nilai x

f '(x) = turunan dari *f(x)*

Pada penelitian ini menggunakan fungsi aktivasi sigmoid biner yang ditunjukkan pada Persamaan (2-1) dan turuanan fungsi sigmoid biner pada Persamaan (2-2).

Penggunaan fungsi aktivasi sigmoid biner pada penelitian ini dikarenakan dalam metode *resilient backpropagation*, fungsi aktivasi yang dipakai harus memenuhi beberapa syarat, yaitu : kontinu, terdeferensial dengan mudah dan merupakan fungsi yang tidak naik turun. Salah satu fungsi aktivasi yang memenuhi tiga syarat tersebut sehingga sering dipakai adalah fungsi aktivasi sigmoid biner dengan range [0,1] [SIA-05].

2.2.2.6 Inisialisasi Bobot dan Bias

Bobot dan bias awal akan mempengaruhi apakah jaringan mencapai titik minimum lokal atau global, dan seberapa cepat konvergensinya. Perubahan bobot antara kedua unit tergantung pada kedua turunan fungsi aktivasi unit di atas dan unit di bawahnya. Pemilihan bobot dan bias awal yang akan menyebabkan fungsi aktivasi atau turunannya menjadi nol sangatlah penting untuk dihindari. Apabila nilai bobot awal terlalu besar akan menyebabkan turunan fungsi sigmoid memiliki nilai sangat kecil, sehingga perubahan bobotnya menjadi sangat kecil. Demikian pula, apabila nilai bobot terlalu kecil akan menyebabkan nilai turunan fungsi sigmoid mendekati nol, sehingga menyebabkan proses pelatihan menjadi sangat lambat [SUP-12].

Nguyen dan Widrow (1990) mengusulkan cara membuat inisialisasi bobot dan bias ke unit tersembunyi sehingga menghasilkan iterasi yang lebih cepat. Algoritma inisialisasi Nguyen-Widrow didefinisikan sebagai persamaan berikut [SIA-05] :

1. Hitung harga faktor penskalaan β dengan Persamaan (2-5) berikut ini :

$$\beta = 0.7\left(\sqrt[n]{p}\right) \tag{2-5}$$

   Di mana :

   β = faktor skala

   n = jumlah unit masukan

   p = jumlah unit tersembunyi

2. Inisialisasi semua bobot ($vij_{(lama)}$) dengan menggunakan bilangan acak pada interval [-0.05,0.05].
3. Hitung nilai $\|v_j\|$ yang dinyatakan dalam Persamaan (2-6) berikut:

$$\left\|v_j\right\| = \sqrt{v^2{}_{1j} + v^2{}_{2j} + ... + v^2{}_{nj}} \tag{2-6}$$

   Di mana :

   v_{nj} = bobot lama antara *input layer* dengan *hidden layer*

   v_j = hasil dari akar penjumlahan v_{nj} kuadrat

4. Bobot yang dipakai sebagai inisialisasi dinyatakan dalam Persamaan (2.7) berikut ini :

$$v_{ij} = \frac{\beta . v_{ij\,(lama)}}{\left\| v_j \right\|} \quad (2\text{-}7)$$

Di mana :

β = faktor skala yang dihasilkan dari Persamaan (2-5)

$vij_{(lama)}$ = bobot awal inisialisasi antara *input layer* dengan *hidden layer*

v_j = hasil dari akar penjumlahan v_{nj} kuadrat pada Persamaan (2-6)

v_{ij} = bobot baru antara *input layer* dengan *hidden layer*

5. Bias yang dipakai sebagai inisialisasi v_{0j} = bilangan acak antara $-\beta$ dan β.

2.2.2.7 Jumlah Unit Tersembunyi

Tidak ada alasan secara teoritis yang menyatakan untuk menggunakan lebih dari dua lapisan tersembunyi. *Resilient backpropagation* dengan sebuah *hidden layer* sudah cukup untuk mengenali sembarang perkawanan antara masukan dan target dengan tingkat ketelitian yang ditentukan [SIA-05].

Penentuan jumlah neuron *hidden layer* dapat menggunakan Persamaan (2-8) sebagai berikut [PAN-12] :

$$neuron_{hidden} = (\frac{2}{3} \times neuron_{input}) + neuron_{output} \quad (2\text{-}8)$$

di mana, $neuron_{hidden}$ adalah jumlah neuron pada *hidden layer*, $neuron_{input}$ adalah jumlah neuron pada *input layer* (jumlah inputan), dan $neuron_{output}$ adalah jumlah neuron pada *output layer* (hasil klasifikasi).

Jumlah unit tersembunyi dapat ditentukan dengan melakukan beberapa percobaan (*trial and error*). Untuk keperluan pengujian jaringan pada penelitian ini akan menggunakan *trial and error*.

Semakin banyak jumlah lapisan tersembunyi dapat menyebabkan jaringan mampu menangani jangkauan statistik yang lebih luas dan tinggi, namun jumlah lapisan yang terlalu banyak juga bisa menimbulkan laju konvergensi menjadi lebih lambat. [SUP-12].

2.2.2.8 Lama Iterasi

Penggunaan *resilient backpropagation* adalah mendapatkan keseimbangan antara pengenalan pola pelatihan secara benar dan respon yang baik untuk pola lain yang sejenis (disebut pengujian). Jaringan dapat dilatih terus menerus hingga semua pola pelatihan dikenali dengan benar. Akan tetapi hal itu tidak menjamin jaringan akan mampu mengenali pola pengujian dengan tepat.

Umumnya data dibagi menjadi 2 bagian, yaitu pola data yang dipakai sebagai pelatihan dan data untuk pengujian. Perubahan bobot dilakukan berdasarkan pola pelatihan seiap kali *epoch*. Akan tetapi selama pelatihan (misal setiap 10 *epoch*), kesalahan yang terjadi dihitung berdasarkan semua data (pelatihan dan pengujian). Selama kesalahan ini menurun, pelatihan terus dijalankan. Akan tetapi jika kesalahannya sudah meningkat, pelatihan tidak ada gunanya untuk diteruskan lagi [SIA – 05].

2.2.3 Metode *Backpropagation*

Backpropagation merupakan salah satu metode atau algoritma pembelajaran dengan supervisi (terbimbing) pada jaringan syaraf tiruan dengan menggunakan arsitektur jaringan *multilayer* atau lapisan banyak. Pada jaringan ini diberikan sepasang pola yang terdiri atas pola masukan dan pola yang diinginkan [KUS-04].

Pada dasarnya terdapat tiga tahap yang harus dilakukan dalam pelatihan jaringan syaraf tiruan menggunakan algoritma *backpropagation,* yaitu tahap pertama adalah tahap perambatan maju di mana pola masukan dihitung maju mulai dari lapisan masukan hingga lapisan keluaran menggunakan fungsi aktivasi yang ditentukan. Tahap kedua adalah tahap perambatan balik di mana selisih antara keluaran jaringan dengan target yang diinginkan merupakan kesalahan yang terjadi. Kemudian, kesalahan tersebut dipropagasikan balik, dimulai dari garis yang berhubungan langsung dengan unit-unit di layar keluaran. Tahap ketiga adalah perubahan bobot dan bias di mana tahap ini dilakukan untuk menurunkan bobot yang terjadi [SUS-12].

2.2.4 Metode Pembelajaran *Resilient Backpropagation*

2.2.4.1 Konsep Metode Pembelajaran *Resilient Backpropagation*

Jaringan syaraf tiruan RPROP (*Resilient Backpropagation*) merupakan algoritma pembelajaran yang dikembangkan dengan tujuan untuk mengatasi kelemahan pada JST backpropagation yang biasanya menggunakan fungsi aktivasi sigmoid. Fungsi aktivasi sigmoid akan membawa input dengan *range* yang tak terbatas ke nilai output dengan *range* yang terbatas, yaitu antara 0 sampai 1. Adapun kelemahan dari metode *backpropagation* adalah penggunaan fungsi aktivasi sigmoid ketika *gradient* akan mendekati nol apabila input yang diberikan sangat banyak, sehingga *gradient* yang bernilai nol akan menyebabkan rendahnya perubahan bobot. Apabila bobot-bobot tidak cukup mengalami perubahan, maka algoritma akan sangat lambat untuk mendekati nilai optimumnya [FEB-07].

Resilient backpropagation adalah adalah algoritma pembelajaran yang bertujuan untuk menghilangkan pengaruh *gradient* terhadap perubahan bobot dengan hanya menggunakan tanda turunannya saja [FAD-09]. Hanya tanda turunan atau *gradient* yang digunakan untuk menunjukkan arah perbaikan bobot.

Sama halnya seperti algoritma *backpropagation gradient descent*, algoritma RPROP juga melakukan dua tahap pelatihan, yakni tahap perambatan maju (*forward*) dan tahap perambatan mundur (*backward*). Tahap perambatan maju digunakan untuk mendapatkan *error output*, sedangkan tahap perambatan mundur digunakan untuk mengubah nilai bobot-bobot [FEB-07].

Perubahan bobot dan bias pada algoritma *resilient backpropagation* dilakukan sesuai dengan perilaku *gradient* di setiap *epoch* pelatihan, sehingga jumlah *epoch* yang diperlukan untuk mencapai target yang diinginkan jauh lebih sedikit [VIR-13].

Beberapa penelitian menggunakan parameter yang disetting secara *default* dalam algoritma *resilient backpropagation*, yaitu penyesuaian minimum (Δ_{min}), penyesuaian maksimum (Δ_{max}), faktor penaik (η^+), dan faktor penurun (η^-). Nilai yang digunakan pada parameter faktor penaik (η^+) adalah 1.2 dan faktor penurun (η^-) adalah 0.5. Pada parameter penyesuaian minimum (Δ_{min}), penyesuaian

maksimum (Δ_{max}) nilai yang digunakan masing-masing sebesar 0.000001 dan 50 [RIE-93].

Menurut Ingvarson (2007) aturan nilai *learning rate* pada algoritma *resilient backpropagation* dapat dilihat pada Persamaan (2-20) sebagai berikut :

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+} \cdot \Delta_{ij}^{(t-1)}, jika \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ \eta^{-} \cdot \Delta_{ij}^{(t-1)}, jika \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)}, selainnya \end{cases} \quad (2\text{-}9)$$

Apabila *gradient* pada iterasi saat ini mempunyai tanda yang sama dengan *gradient* sebelumnya, maka nilai *learning rate* (nilai penyesuaian) ditingkatkan dengan faktor penaik dan juga sebaliknya. Menurut Ingvarson (2007) aturan penyesuaian bobot dapat dilihat pada Persamaan (2-20) sebagai berikut :

$$\Delta w_{ij}^{(t)} = \begin{cases} + \Delta_{ij}^{(t)}, jika \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ - \Delta_{ij}^{(t)}, jika \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ 0, selainnya \end{cases} \quad (2\text{-}10)$$

Aturan untuk mendapatkan bobot baru dapat dilihat pada Persamaan (2-11) sebagai berikut :

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \quad (2\text{-}11)$$

Di mana :

η^{+} = faktor penaik (*learning rate* penaik)

η^{-} = faktor penurun (*learning rate* penurun)

$\frac{\partial E}{\partial w_{ij}}^{(t)}$ = kemiringan kurva *error* terhadap bobot (*gradient*) pada iterasi saat ini.

$\frac{\partial E}{\partial w_{ij}}^{(t-1)}$ = kemiringan kurva *error* terhadap bobot (*gradient*) pada iterasi sebelumnya.

$\Delta_{ij}^{(t-1)}$ = *learning rate* indeks ke-ij pada iterasi saat ini.

$\Delta_{ij}^{(t-1)}$ = *learning rate* indeks ke-ij pada iterasi sebelumnya.

$\Delta w_{ij}^{(t)}$ = perubahan bobot saat ini

$w_{ij}^{(t)}$ = bobot saat ini

$w_{ij}^{(t+1)}$ = bobot baru setelah diperbarui

Berdasarkan pada penelitian yang sudah ada sebelumnya, pada penelitian ini akan digunakan nilai *default* pada keempat parameter tersebut sesuai dengan penelitian yang telah ada sebelumnya, yaitu nilai penyesuaian minimum (Δ_{min}) sebesar 0.000001, penyesuaian maksimum (Δ_{max}) sebesar 50, faktor penaik (η^+) sebesar 1.2, dan faktor penurun (η^-) sebesar 0.5.

Pelatihan dilakukan berulang-ulang dan berhenti jika telah mencapai batas *epoch* maksimum yang ditentukan dan nilai *error* kurang dari *Mean Square Error* (MSE). Ketepatan algoritma *resilient backpropagation* ditentukan dengan *Mean Square Error* (MSE). Semakin kecil nilai MSE, maka arsitektur jaringan akan semakin baik [FAU-94]. Nilai MSE dapat dihitung dengan Persamaan (2-12) sebagai berikut :

$$MSE = \frac{\sum_p (t_p - y_p)^2}{N * K} \qquad (2\text{-}12)$$

Di mana :

p = jumlah *neuron* pada *output layer*

t = target

y = output

N = jumlah data latih

K = banyaknya keluaran

2.2.4.2 Langkah-langkah Pembelajaran *Resilient Backpropagation*

Proses perambatan maju (*forward*) pada algoritma RPROP sama dengan algoritma propagasi balik umumnya, sedangkan proses perambatan mundur (*backward*) berbeda. Algoritma pembelajaran RPROP berdasarkan aturan yang dikemukakan oleh Ingvarson (2007) adalah sebagai berikut :

1. Inisialisasi bobot

   Bobot awal antara neuron di *input layer* dengan neuron di *hidden layer* diinisialisasi dengan algoritma Nguyen Widrow. Bobot antara bias *layer* dengan neuron di *hidden layer*, bobot antara neuron di *hidden layer* dengan neuron di *output layer*, serta bobot antara bias *layer* dengan neuron di *output layer* ditentukan secara acak dengan nilai sekecil mungkin, misalkan [-0.05, 0.05].
2. Inisialisasi *error target*, jumlah neuron pada *hidden layer*, *learning rate*, faktor penaik, faktor penurun, penyesuaian minimum, penyesuaian maksimum dan jumlah maksimum *epoch*.
3. Selama (*epoch* < maksimum *epoch*) atau (MSE > *error target*), maka:
   a. Lakukan *feed forward* untuk setiap pasangan pelatihan:
      1. Tiap unit masukan (x_i, i=1,…,n) menerima sinyal masukan x_i dan meneruskan sinyal tersebut ke semua unit pada *layer* di atasnya (*hidden layer*)
      2. Tiap-tiap unit pada *hidden layer* (z_j, j=1,….,p) menjumlahkan sinyal–sinyal input terbobot,

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_{i.}.v_{ij} \qquad (2\text{-}13)$$

         lalu menghitung sinyal keluarannya dengan fungsi aktivasi,

$$z_j = f(z\_in_j) \qquad (2\text{-}14)$$

         dan mengirimkan sinyal ini ke seluruh unit pada lapisan atasnya (lapisan *output*).
      3. Setiap unit output (y_k, k=1,…,m) menghitung total sinyal masukan terbobot,

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} w_{jk}.z_j \qquad (2\text{-}15)$$

         lalu menghitung sinyal keluaran dengan fungsi aktivasi,

$$y_k = f(y\_in_k) \qquad (2\text{-}16)$$

b. Lakukan *resilient backpropagation* untuk masing-masing pasangan pelatihan:

1. Menghitung nilai informasi *error* (δ_k) pada tiap-tiap unit output :

$$\delta_k = f'(y_k)\frac{1}{2}(t_k - y_k) \quad (2\text{-}17)$$

lalu hitung nilai informasi *error* (δ_j) pada tiap-tiap unit *hidden* dengan melakukan proses *backpropagation* terhadap *error* pada output (δ_k)

$$\delta_j = f'(z_j)\sum_k \delta_k w_{jk} \quad (2\text{-}18)$$

2. Menghitung nilai *gradient* $\left(\frac{\partial E}{\partial v_{ij}}^{(t)}\right)$ dan $\left(\frac{\partial E}{\partial w_{jk}}^{(t)}\right)$, yaitu dengan cara : menghitung nilai turunan pertama fungsi *error* terhadap bobot pada *input* ke *hidden layer* $\left(\frac{\partial E}{\partial v_{ij}}^{(t)}\right)$

$$\frac{\partial E}{\partial v_{ij}}^{(t)} = \frac{\partial E}{\partial v_{ij}}^{(t)} + \delta_j x_i \quad (2\text{-}19)$$

dan menghitung nilai turunan pertama fungsi *error* terhadap bobot pada *hidden* ke *output layer* $\left(\frac{\partial E}{\partial w_{jk}}^{(t)}\right)$

$$\frac{\partial E}{\partial w_{jk}}^{(t)} = \frac{\partial E}{\partial w_{jk}}^{(t)} + \delta_k z_j \quad (2\text{-}20)$$

c. Perbarui Bobot (berlaku untuk semua bobot)

1. Menghitung nilai *learning rate* yang baru pada semua bobot

Jika $\frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} > 0$ maka

$$\Delta_{ij}^{(t)} = \min(\Delta_{ij}^{(t-1)} * \eta^+, \Delta_{\max})$$

Jika $\frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0$ maka

$$\Delta_{ij}^{(t)} = \max(\Delta_{ij}^{(t-1)} * \eta^-, \Delta_{\min})$$

$$\frac{\partial E}{\partial w_{ij}}^{(t)} = 0$$

Jika tidak, maka

$$\Delta_{ij}^{(t)} = \Delta_{ij}^{(t-1)} \quad (2\text{-}21)$$

2. Menghitung nilai perubahan bobot

Jika $\frac{\partial E}{\partial w_{ij}}^{(t)} > 0$ maka

$$\Delta w_{ij}^{(t)} = +\Delta_{ij}^{(t)}$$

Jika $\frac{\partial E}{\partial w_{ij}}^{(t)} < 0$ maka

$$\Delta w_{ij}^{(t)} = -\Delta_{ij}^{(t)}$$

Jika tidak, maka

$$\Delta w_{ij}^{(t)} = 0 \quad (2\text{-}22)$$

3. Menghitung nilai bobot baru

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \quad (2\text{-}23)$$

Jika *epoch* telah maksimum atau nilai MSE < *error target*, maka berhenti.

Keterangan dari simbol :

i = indeks untuk unit *input*

j = indeks untuk unit *hidden*

k = indeks untuk unit *output*

t_k = target

x_i = unit *input i*

v_{oj} = bobot pada bias *layer* ke *hidden layer j*

v_{ij} = bobot dari unit *input i* menuju ke unit *hidden j*

z_j = unit *hidden j*

z_in_j = *input* jaringan ke z_j

w_{ok} = bobot pada bias *layer* ke *output layer k*

w_{jk} = bobot dari unit *hidden j* menuju ke unit *output k*

y_k = unit *output k*

y_in_k = *input* jaringan ke y_k

δ_k = informasi *error* pada unit *output* y_k ke unit *hidden*

δ_j = informasi *error* dari *output layer* ke unit hidden z_j

η^+ = faktor penaik

η^- = faktor penurun

Δ_{min} = penyesuaian minimum

Δ_{max} = penyesuaian maksimum

E = *error* tiap pasangan

$\frac{\partial E}{\partial w_{ij}}^{(t)}$ = kemiringan kurva *error* terhadap bobot (*gradient*) pada iterasi saat ini.

$\frac{\partial E}{\partial w_{ij}}^{(t-1)}$ = kemiringan kurva *error* terhadap bobot (*gradient*) pada iterasi sebelumnya.

$\Delta_{ij}^{(t-1)}$ = *learning rate* indeks ke-ij pada iterasi saat ini.

$\Delta_{ij}^{(t-1)}$ = *learning rate* indeks ke-ij pada iterasi sebelumnya.

$\Delta w_{ij}^{(t)}$ = perubahan bobot saat ini

$w_{ij}^{(t)}$ = bobot saat ini

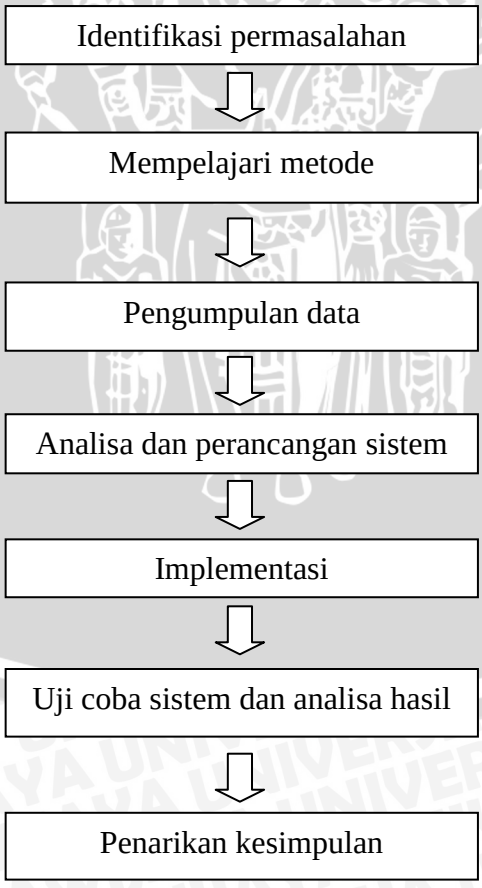
$w_{ij}^{(t+1)}$ = bobot baru setelah diperbarui

BAB III
METODE PENELITIAN DAN PERANCANGAN

3.1 Metode Penelitian

Pada subbab metodologi penelitian ini akan dibahas mengenai tahapan yang merepresentasikan langkah-langkah yang akan dilakukan peneliti dalam proses penelitian yang berjudul "Implementasi Jaringan Syaraf Tiruan dengan Metode Pembelajaran *Resilient Backpropagation* (RPROP) untuk Klasifikasi Status Tahapan Keluarga Sejahtera".

Tahapan penelitian yang menunjukkan langkah-langkah yang akan digunakan untuk memecahkan permasalahan klasifikasi status tahapan keluarga sejahtera menggunakan jaringan syaraf tiruan *resilient backpropagation* dapat diilustrasikan pada Gambar 3.1.



Gambar 3. 1 Blog diagram penelitian

Tahap-tahap penelitian yang akan dilakukan sebagai berikut:

1. Identifikasi permasalahan pada kasus klasifikasi status tahapan keluarga sejahtera.
2. Mempelajari literatur atau sumber-sumber yang terkait dengan metode yang digunakan (*resilient backropagation*) dan objek penelitian (status tahapan keluarga sejahtera).
3. Pengumpulan data berupa indikator-indikator yang mendukung proses klasifikasi status tahapan keluarga sejahtera dan data hasil pendataan keluarga tahun 2013 di desa Tulungrejo RT. 01, 02, 03, 04, 05 - RW. 05 yang didapatkan dari Posyandu Kecamatan Bumi Aji, Kota Batu.
4. Menganalisa dan merancang sistem dengan menggunakan hasil pembelajaran/pelatihan pada tahap sebelumnya.
5. Membuat sistem berdasarkan analisis dan perancangan yang telah dilakukan.
6. Melakukan uji coba terhadap sistem dan menganalisa hasil dari uji coba.
7. Menarik kesimpulan dari hasil uji coba sistem yang telah dilakukan berdasarkan rumusan masalah.

3.1.1 Identifikasi Permasalahan

Identifikasi permasalahan adalah mengenali masalah dengan cara mendaftar faktor-faktor yang berupa permasalahan, sehingga masalah yang telah dipilih memiliki nilai yang sangat penting untuk dipecahkan [SET-12].

Identifikasi masalah sebenarnya dilakukan untuk menemukan ruang lingkup masalah tertentu, kemudian dipilih salah satu masalah yang sesuai dengan kemampuan peneliti, baik dari segi pelaksanaan ataupun kurikulumnya [TAH-12].

Masalah yang diidentifikasi dalam penelitian ini adalah kasus klasifikasi status tahapan keluarga sejahtera. Status tahapan keluarga sejahtera erat kaitannya dengan permasalahan pembangunan kependudukan di Indonesia. Dengan adanya pengklasifikasian status tahapan keluarga sejahtera, maka dapat membantu pemerintah untuk mewujudkan kesuksesan program pembangunan keluarga sejahtera, karena dengan adanya pengelompokkan status keluarga pada suatu daerah dapat diketahui tingkat kesejahteraanya, terutama dari segi ekonomi. Hal

ini dapat membantu pemerintah dalam menuntaskan keluarga yang statusnya masih tertinggal untuk menuju status yang lebih sejahtera.

Pengklasifikasian status tahapan keluarga sejahtera yang dilakukan oleh para kader atau petugas setempat (Posyandu kecamatan Bumiaji, Kota Batu) masih dilakukan secara manual, sehingga masih menyulitkan petugas. Petugas harus mempertimbangkan 21 nilai indikator yang telah diperoleh dari masing-masing keluarga. Jumlah keluarga dalam suatu wilayah desa atau kelurahan tidaklah sedikit, tentunya hal ini akan memakan waktu yang cukup lama bila proses pengklasifikasian dilakukan secara manual. Sebenarnya sudah tersedia aplikasi khusus untuk pengelolaan pemutakhiran data keluarga dari pemerintah yang beralamat pada http://aplikasi.bkkbn.go.id/. Namun, aplikasi ini belum mendukung pengklasifikasian status tahapan keluarga sejahtera secara otomatis, sehingga klasifikasi yang dihasilkan belum terjamin keakuratannya. Aplikasi ini hanya mendukung fitur untuk penyimpanan data keluarga secara online.

Berdasarkan uraian permasalahan di atas, maka dapat diidentifikasi beberapa masalah yang akan dipecahkan dalam kasus klasifikasi status tahapan keluarga sejahtera, yaitu :

1. Kesulitan untuk melakukan klasifikasi status tahapan keluarga secara manual, karena banyaknya jumlah indikator yang harus dipertimbangkan (21 indikator).
2. Keakuratan hasil klasifikasi status tahapan keluarga secara manual masih belum terjamin.

Berdasarkan masalah yang telah teridentifikasi pada ruang lingkupnya, maka peneliti menentukan metode yang akan dilakukan untuk menyelesaikan permasalahan tersebut. Peneliti memutuskan untuk menggunakan metode jaringan syaraf tiruan *resilient backpropagation* untuk menyelesaikan permasalahan ini, sehingga diperoleh judul "Implementasi Jaringan Syaraf Tiruan dengan Metode Pembelajaran RPROP (*Resilient Backpropagation*) untuk Klasifikasi Status Tahapan Keluarga Sejahtera".

3.1.2 Mempelajari Metode

Tahap mempelajari metode di sini bisa disebut juga dengan studi literatur. Studi literatur adalah penelusuran literatur yang bersumber dari buku, media, pakar ataupun dari hasil penelitian orang lain yang bertujuan untuk menemukan pemecahan masalah dan menyusun dasar teori yang akan digunakan [JAY-12].

Setelah dilakukan identifikasi permasalahan, maka peneliti mempelajari metode yang telah ditentukan, yaitu jaringan syaraf tiruan *resilient backpropagation*. Metode yang dipelajari ini bersumber dari jurnal-jurnal ilmiah, paper, buku, situs-situs ilmiah, dan penelitian yang sudah ada sebelumnya. Cara ini dilakukan untuk mendapatkan dasar-dasar referensi yang kuat bagi peneliti guna menyelesaikan laporan.

3.1.3 Pengumpulan Data

Pengumpulan data adalah kegiatan yang dilakukan oleh peneliti untuk mengumpulkan sejumlah data lapangan yang diperlukan untuk menjawab pertanyaan penelitian (untuk penelitian kualitatif) atau menguji hipotesis (untuk penelitian kuantitatif) [SET-12].

Pengumpulan data merupakan kegiatan yang perlu dilakukan oleh peneliti untuk menunjang keberhasilan dari penelitian. Adapun teknik pengumpulan data, jenis data, dan sumber data yang dikumpulkan adalah sebagai berikut :

a. Teknik pengumpulan data

   Pengumpulan data dalam suatu penelitian dapat dilakukan dengan teknik wawancara, observasi, kuisioner, dokumentasi ataupun triangulasi (gabungan) [SUR-10].

   Teknik pengumpulan data yang digunakan dalam penelitian ini adalah dokumentasi. Dokumen diperoleh dari narasumber berupa data *hardcopy*. Kemudian peneliti memasukkan semua data yang diperoleh ke dalam format Microsoft Excel 2010.

b. Jenis data

   Jenis data berdasarkan cara memperolehnya dapat dikategorikan menjadi dua, yaitu data primer dan data sekunder. Data primer adalah data yang diperoleh secara langsung dari objek penelitian oleh peneliti. Data

sekunder adalah data yang diperoleh tidak secara langsung dari objek penelitian atau peneliti mendapatkan data yang sudah jadi yang dikumpulkan oleh pihak lain dengan berbagai cara [SUS-12].

Jenis data yang dikumpulkan dalam penelitian ini termasuk data sekunder, karena diperoleh dari pihak lain yang kemudian diserahkan kepada peneliti. Data yang diperoleh berupa dokumen *hardcopy* hasil pendataan keluarga.

c. Sumber data

Sumber data dapat digolongkan menjadi tiga bagian, yaitu data internal, data eksternal, dan data ekstraksi. Pertama, data internal adalah data yang menggambarkan situasi dan kondisi pada suatu organisasi secara internal. Kedua, data eksternal adalah data yang menggambarkan situasi serta kondisi yang ada di luar organisasi. Ketiga, data ekstraksi merupakan penggabungan dari data internal dan data eksternal [DAN-12].

Sumber data dalam penelitian ini adalah data internal saja. Pada penelitian ini yang menjadi data internal adalah 21 indikator status tahapan keluarga sejahtera dan data nilai dari indikator tersebut pada masing-masing keluarga. Data yang diperoleh berupa dokumen pendataan keluarga. Dokumen tersebut berisi data hasil pemutakhiran keluarga tahun 2013 di Desa/Kelurahan Tulungrejo, RT. 01, 02, 03, 04, 05 - RW. 05, Kecamatan Bumi Aji, Kota Batu. Data tersebut diambil dari Posyandu Kecamatan Bumiaji, Kota Batu.

3.1.4 Analisa dan Perancangan Sistem

Analisa sistem adalah kegiatan menganalisa sistem yang ada berdasarkan data dan informasi yang telah dikumpulkan sebelumnya, diharapkan pada peneliti mengetahui dan memahami sistem yang sedang berjalan, sehingga bisa menjadi acuan pada perancangan sistem yang baru [DAN-12].

Pada tahap analisa sistem, peneliti akan melakukan analisa kebutuhan sistem, analisa perancangan basis data, dan analisa perancangan antarmuka. Setelah itu, akan dilakukan perancangan sistem sesuai analisa yang telah dilakukan.

Perancangan sistem adalah penyusunan proses, data, aliran proses dan hubungan antar data yang dapat memenuhi kebutuhan pihak yang terkait sesuai dengan analisa kebutuhan. Hasil yang diharapkan pada perancangan sistem adalah bisa merancang sebuah sistem berdasarkan hasil dari analisa sistem yang ada [DAN-12].

Perancangan sistem yang akan dilakukan untuk menyelesaikan masalah klasifikasi status tahapan keluarga sejahtera, meliputi deskripsi umum sistem, perancangan model JST, perancangan proses JST *resilient backpropagation* dengan menggunakan *flowchart*, contoh perhitungan manual, perancangan basis data, perancangan antarmuka, dan perancangan uji coba sistem yang akan disajikan dalam bentuk tabel pengujian.

3.1.5 Implementasi

Implementasi merupakan sekumpulan aktivitas di mana rancangan perangkat lunak telah dibuat pada tahap perancangan kemudian dikodekan ke dalam bentuk kode program menggunakan bahasa pemrograman tertentu agar dapat dijalankan pada komputer atau perangkat lainnya yang dikehendaki [DAN-12].

Pada tahapan implementasi ini akan dibuat pengkodean program sistem klasifikasi status tahapan keluarga sejahtera dengan JST *resilient backpropagation* (RPROP) berdasarkan rancangan yang telah dibuat. Bahasa pemrograman yang digunakan adalah PHP dengan editor Codelobster PHP Edition versi 4.2.1.

3.1.6 Uji Coba Sistem dan Analisa Hasil

Setelah melakukan proses implementasi, maka proses selanjutnya adalah uji coba dengan tujuan untuk mengetahui bahwa sistem yang dibuat telah sesuai dengan kebutuhan serta menguji metode yang diterapkan pada sistem untuk menyelesaikan permasalahan [MAN-10].

Uji coba dilakukan dengan cara melakukan pelatihan pada jaringan syaraf tiruan dengan menggunakan 2 parameter JST, yaitu jumlah neuron pada *hidden layer* dan nilai *learning rate*. Uji coba dilakukan guna menentukan jumlah neuron pada *hidden layer* dan nilai *learning rate* yang optimal, sehingga dapat diambil

sebuah konfigurasi jaringan yang terbaik untuk digunakan dalam mengklasifikasikan status tahapan keluarga sejahtera.

Konfigurasi jaringan *resilient backpropagation* yang dihasilkan perlu diuji untuk mengetahui kemampuannya pada saat mempelajari data latih yang diberikan. Pengujian dilakukan dengan menggunakan data set yang telah dilatihkan dengan tujuan untuk melihat kinerja sistem aplikasi pada kasus klasifikasi status tahapan keluarga sejahtera, sehingga diperoleh informasi mengenai tingkat akurasi. Selanjutnya dapat dilakukan analisa terhadap hasil uji coba sistem yang dilakukan.

3.1.7 Penarikan Kesimpulan

Penarikan kesimpulan adalah berisi kesimpulan yang dapat diambil dari pembahasan pada pengujian dan analisis, sehingga kesimpulan bisa sesuai dengan rumusan masalah [BAS-11].

Setelah dilakukan analisa terhadap hasil pengujian, maka akan dilakukan penarikan kesimpulan terhadap permasalahan klasifikasi status tahapan keliarga sejahtera dengan metode pembelajaran RPROP (*Resilient Backpropagation*). Kesimpulan berisi jawaban terhadap rumusan masalah yang ada.

3.2 Perancangan Sistem

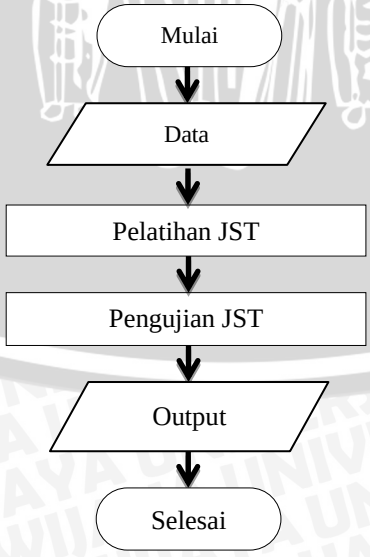
Pada subbab perancangan sistem ini akan dibahas mengenai deskripsi umum sistem, analisa kebutuhan sistem, perancangan model JST, perancangan proses JST *resilient backpropagation*, contoh perhitungan manual, perancangan basis data, perancangan antarmuka, dan perancangan uji coba sistem.

3.2.1 Deskripsi Umum Sistem

Sistem yang akan dibuat dalam skripsi ini bertujuan untuk mengimplementasikan dan mengetahui kinerja dari metode jaringan syaraf tiruan *resilient backpropagation* dalam permasalahan klasifikasi status tahapan keluarga sejahtera. Pengklasifikasian dibagi menjadi 5 kelas, yaitu tahap keluarga prasejahtera, keluarga sejahtera I, keluarga sejahtera II, keluarga sejahtera III, dan keluarga sejahtera III plus.

Dalam mengenali pola pengklasifikasian status tahapan keluarga sejahtera, sistem membutuhkan masukan berupa nilai dari 21 indikator yang telah dimasukkan oleh user. Sistem akan mengolah *input* yang diberikan oleh user dengan cara melakukan pelatihan atau pembelajaran. Pada proses pelatihan menggunakan algoritma *feedforward*, kemudian akan dilakukan proses resilient backpropagation pada perhitungan nilai informasi *error* dan nilai *gradien*t. Apabila nilai MSE belum terpenuhi sesuai dengan *error target* atau selama *epoch* masih belum maksimum *epoch*, maka akan dilakukan proses *resilient backpropagation*, mulai dari *feedforward* sampai perbaikan bobot. Apabila nilai *error* telah mencapai nilai minimum atau *epoch* telah mencapai maksimum, maka bobot terakhir yang telah diperbarui akan disimpan. Selanjutnya akan dilakukan proses pengujian sistem, yaitu dengan menerapkan algoritma *feedforward* dari algoritma pelatihan. Selama proses pengujian, bobot-bobot yang dipakai adalah bobot-bobot akhir yang diperoleh dari proses pelatihan. Apabila proses pengujian telah selesai, maka akan dihasilkan *output* berupa hasil pengklasifikasian status tahapan keluarga sejahtera.

Secara keseluruhan, tahap pengembangan sistem klasifikasi status tahapan keluarga sejahtera menggunakan jaringan syaraf tiruan *resilient backpropagation* dapat diilustrasikan pada Gambar 3.2.



Gambar 3. 2 Diagram alir pengembangan sistem secara umum

3.2.2 Analisa Kebutuhan Sistem

Berdasarkan literatur-literatur yang dipelajari tentang metode pembelajaran *resilient backpropagation* (RPROP) pada kasus klasifikasi, maka peneliti membagi analisa kebutuhan sistem menjadi 3 bagian, yaitu analisa kebutuhan proses, analisa kebutuhan perancangan basis data, dan analisa kebutuhan perancangan antarmuka.

3.2.3 Analisa Kebutuhan Proses

Kebutuhan proses dalam sistem klasifikasi status tahapan keluarga sejahtera menggunakan metode pembelajaran JST *resilient backpropagation* (RPROP) antara lain :

1. Proses pelatihan JST *resilient backpropagation* terhadap data klasifikasi status tahapan keluarga sejahtera.
2. Proses inisialisasi bobot dan bias awal.
3. Proses *feedforward*, yaitu proses pelatihan JST pada tahap perambatan maju.
4. Proses info error *resilient backpropagation*, yaitu proses pelatihan JST pada tahap perhitungan nilai informasi *error* pada masing-masing lapisan .
5. Proses hitung *gradient*, yaitu proses pelatihan JST pada tahap perhitungan nilai *gradient* pada masing-masing lapisan yang nantinya digunakan untuk perbaikan bobot.
6. Proses perbaikan bobot, meliputi bobot antara *hidden layer* ke *output layer* (w_{jk}), bobot antara bias ke *output layer* (w_{0k}), bobot antara *input layer* ke *hidden layer* (v_{ij}), dan bobot antara bias ke *hidden layer* (v_{0j}).
7. Proses pengujian JST dengan menerapkan tahap perambatan maju (*feedforward*), sehingga dihasilkan *output* berupa hasil klasifikasi status tahapan keluarga sejahtera.

3.2.3.1 Analisa Kebutuhan Perancangan Basis Data

Sistem yang akan dibuat memerlukan database untuk menyimpan seluruh data yang digunakan oleh sistem. Database yang digunakan dalam implementasi sistem ini adalah MySQL versi server 5.5.16 dengan menggunakan phpMyAdmin versi 3.4.5. Tabel yang dapat disusun berdasarkan kebutuhan sistem ini, yaitu:

1. Tabel akun, yaitu tabel untuk menyimpan mengenai informasi *username, password* dan status user sebagai administrator untuk keperluan login ke dalam halaman admin.
2. Tabel detail_akun, yaitu tabel untuk menyimpan data rincian akun user sebagai administrator.
3. Tabel parameter, yaitu tabel untuk menyimpan data parameter JST yang digunakan untuk tahap pelatihan JST *resilient backpropagation*.
4. Tabel history_pelatihan, yaitu tabel untuk menyimpan data riwayat dari pelatihan JST *resilient backpropagation* yang pernah dilakukan..

3.2.3.2 Analisa Kebutuhan Perancangan Antarmuka

Pada perancangan antarmuka akan digunakan bahasa pemrograman PHP berbasis web dengan Codelobster PHP Edition 4.2.1 sebagai editornya. Antaramuka yang dihasilkan akan dijalankan di *web browser* Google Chrome versi 34.0.1847.131.

Kebutuhan perancangan antarmuka dibagi menjadi dua bagian, yaitu antarmuka utama dan antarmuka administrator.

a. Antarmuka utama, meliputi :
    1. Antarmuka halaman *"Home",* yaitu untuk menampilkan halaman utama sistem.
    2. Antarmuka halaman "Deskripsi Sistem", yaitu menampilkan informasi mengenai deskripsi umum sistem dalam penelitian yang meliputi objek penelitian (klasifikasi status tahapan keluarga sejahtera) dan metode yang digunakan untuk menyelesaikan kasus (JST RPROP), sehingga terbentuklah sistem tersebut.
    3. Antarmuka halaman "Tentang JST RPROP", yaitu menampilkan informasi mengenai metode pembelajaran atau pelatihan JST RPROP *(resilient backpropagation)* yang meliputi keunggulan dan kelemahan metode tersebut.
    4. Antarmuka halaman "Aplikasi", yaitu menampilkan *form* untuk melakukan pengujian terhadap satu kali data masukan, sehingga user tidak perlu login ke sebagai administrator untuk mencoba sistem.

b. Antarmuka halaman Administrator, meliputi :
   1. Antarmuka halaman "Profil", yaitu untuk menampilkan informasi mengenai data akun pribadi administrator.
   2. Antarmuka halaman "Pelatihan JST", yaitu untuk menampilkan *form* pelatihan JST beserta hasil dari pelatihannya.
   3. Antarmuka halaman "Riwayat Pelatihan", yaitu untuk menampilkan riwayat pelatihan JST yang pernah dilakukan dengan menggunakan parameter-parameter yang ditentukan.
   4. Antarmuka halaman "Bobot dan Bias", yaitu untuk menampilkan data bobot dan bias terakhir hasil pelatihan JST yang disimpan ke dalam dataset.
   5. Antarmuka halaman "Pengujian JST", yaitu untuk menampilkan *form* pengujian JST beserta hasil dari pengujiannya.
   6. Antarmuka halaman "Logout", yaitu sebagai sarana keluar dari hak akses sebagai administrator.

3.2.4 Perancangan Model JST

Perancangan model untuk JST *resilient backpropagation* pada klasifikasi status tahapan keluarga sejahtera ini terdiri dari 3 rancangan pokok, yaitu penetapan masukan, penetapan keluaran, dan arsitektur jaringan.

3.2.4.1 Penetapan Masukan

Penetapan masukan dilakukan dengan memasukkan indikator untuk penentuan status tahapan keluarga sejahtera yang berjumlah 21 masukan (x), yaitu:

1. Pada umumnya anggota keluarga makan dua kali sehari atau lebih (x_1).
2. Anggota keluarga memiliki pakaian yang berbeda untuk di rumah, bekerja/sekolah dan bepergian (x_2).
3. Rumah yang ditempati keluarga mempunyai atap, lantai dan dinding yang baik (x_3).
4. Bila ada anggota keluarga sakit dibawa ke sarana kesehatan (x_4).
5. Bila pasangan usia subur ber-KB pergi ke sarana pelayanan kontrasepsi (x_5).

6. Semua anak umur 7-15 tahun dalam keluarga bersekolah (x_6).
7. Pada umumnya anggita keluarga melaksanakan ibadah sesuai dengan agama dan kepercayaannya (x_7).
8. Paling kurang sekali seminggu seluruh keluarga makan daging ikan/telur (x_8).
9. Seluruh anggota keluarga memperoleh paling kurang satu stel pakaian baru dalam setahun (x_9).
10. Luas lantai rumah paling kurang 8 m^2 setiap penghuni rumah (x_{10}).
11. Tiga bulan terakhir keluarga dalam keadaan sehat sehingga dapat melaksanakan tugas/ fungsinya (x_{11}).
12. Ada seorang atau lebih anggota keluarga yang bekerja untuk memperoleh penghasilan (x_{12}).
13. Seluruh anggota keluarga umur 10-60 tahun bisa baca tulis latin (x_{13}).
14. Pasangan usia subur dengan anak 2 atau lebih menggunakan alat/ obat kontrasepsi (x_{14}).
15. Keluarga berupaya meningkatkan pengetahuan agama (x_{15}).
16. Sebagian penghasilan keluarga ditabung dalam bentuk uang maupun barang (x_{16}).
17. Kebiasaan keluarga makan bersama paling kurang seminggu sekali dimanfaatkan untuk berkomunikasi (x_{17}).
18. Keluarga sering ikut dalam kegiatan masyarakat di lingkungan tempat tinggal (x_{18}).
19. Keluarga memperoleh informasi dari surat kabar/ majalah/ radio/ TV (x_{19}).
20. Keluarga secara teratur dengan sukarela memberikan sumbangan materi untuk kegiatan sosial (x_{20}).
21. Ada anggota keluarga yang aktif sebagai pengurus Perkumpulan Sosial/ Yayasan/ Institusi Masyarakat (x_{21}).

Proses penginputan data indikator-indikator di atas diproses ke menjadi inputan JST, yaitu dengan cara mengkonversikan nilai masing-masing indikator ke dalam interval [0, 1] sesuai dengan interval pada fungsi aktivasi sigmoid biner. Berdasarkan aturan pengisian masing-masing indikator pada *form* F/I/MDK/11, maka aturan tersebut adalah sebagai berikut :

1. Diisi dengan tanda centang (v), jika memenuhi indikator.

2. Diisi dengan tanda (-), jika indikator tidak berlaku pada keluarga yang bersangkutan.
3. Diisi dengan tanda (x), jika tidak memenuhi indikator karena alasan bukan ekonomi.
4. Diisi dengan tanda (x*), jika tidak memenuhi indikator karena alasan ekonomi.

Berdasarkan aturan pengisian yang terdapat pada *form* F/I/MDK/11, maka peneliti membagi nilai konversi data input menjadi 4 macam, yaitu tanda (v) dengan nilai 0.9, tanda (-) dengan nilai 0.7, tanda (x) dengan nilai 0.5, dan tanda (x*) dengan nilai 0.3. Tabel konversi nilai data input pada indikator ditunjukkan pada Tabel 3.1.

Tabel 3. 1 Tabel konversi nilai input pada indikator

No.	Tanda	Nilai
1	v	0.9
2	-	0.7
3	x	0.5
4	x*	0.3

Rentang nilai yang akan mewakili status tahapan keluarga sejahtera dalam obyek penelitian ini terdapat 5 kelas kategori status tahapan keluarga sejahtera. Seperti halnya pada proses penginputan data indikator, maka pada input target juga akan dilakukan proses konversi. Konversi nilai menggunakan pendekatan nilai keluaran fungsi aktivasi sigmoid biner, yaitu pada interval [0, 1]. Nilai konversi pada data input target merupakan nilai tengah dari rentang nilai keluaran yang ada. Tabel konversi nilai input target dapat dilihat pada Tabel 3.2.

Tabel 3. 2 Tabel konversi nilai input target

No.	Status Tahapan Keluarga Sejahtera	Range Nilai Keluaran	Nilai Target
1	K Prasejahtera	$0 \leq x < 0.2$	0.1
2	KS I	$0.2 \leq x < 0.4$	0.3
3	KS II	$0.4 \leq x < 0.6$	0.5
4	KS III	$0.6 \leq x < 0.8$	0.7
5	KS III+	$0.8 \leq x < 1$	0.9

Berdasarkan Tabel 3.2, maka dapat direpresentasikan nilai keluaran JST yang dihasilkan pada saat pengujian *(feedforward)* sebagai berikut:

- Apabila nilai keluaran berada pada $0 \leq y_k < 0.2$, maka hasil prediksi klasifikasi tergolong ke dalam kategori "Keluarga Prasejahtera (KPS)".
- Apabila nilai keluaran berada pada $0.2 \leq y_k < 0.4$, maka hasil prediksi klasifikasi tergolong ke dalam kategori "Keluarga Sejahtera I (KS I)".
- Apabila nilai keluaran berada pada $0.4 \leq y_k < 0.6$, maka hasil prediksi klasifikasi tergolong ke dalam kategori "Keluarga Sejahtera II (KS II)".
- Apabila nilai keluaran berada pada $0.6 \leq y_k < 0.8$, maka hasil prediksi klasifikasi tergolong ke dalam kategori "Keluarga Sejahtera III (KS III)".
- Apabila nilai keluaran berada pada $0.8 \leq y_k < 1$, maka hasil prediksi klasifikasi tergolong ke dalam kategori "Keluarga Sejahtera III+ (KS III+)".

3.2.4.2 Penetapan Keluaran

Keluaran atau *output* yang akan dihasilkan pada sistem klasifikasi status tahapan keluarga sejahtera menggunakan JST *resilient backpropagation* ini adalah salah satu dari 5 macam kelas klasifikasi, yaitu :

1. Keluarga Prasejahtera
2. Keluarga Sejahtera I
3. Keluarga Sejahtera II
4. Keluarga Sejahtera III
5. Keluarga Sejahtera III Plus

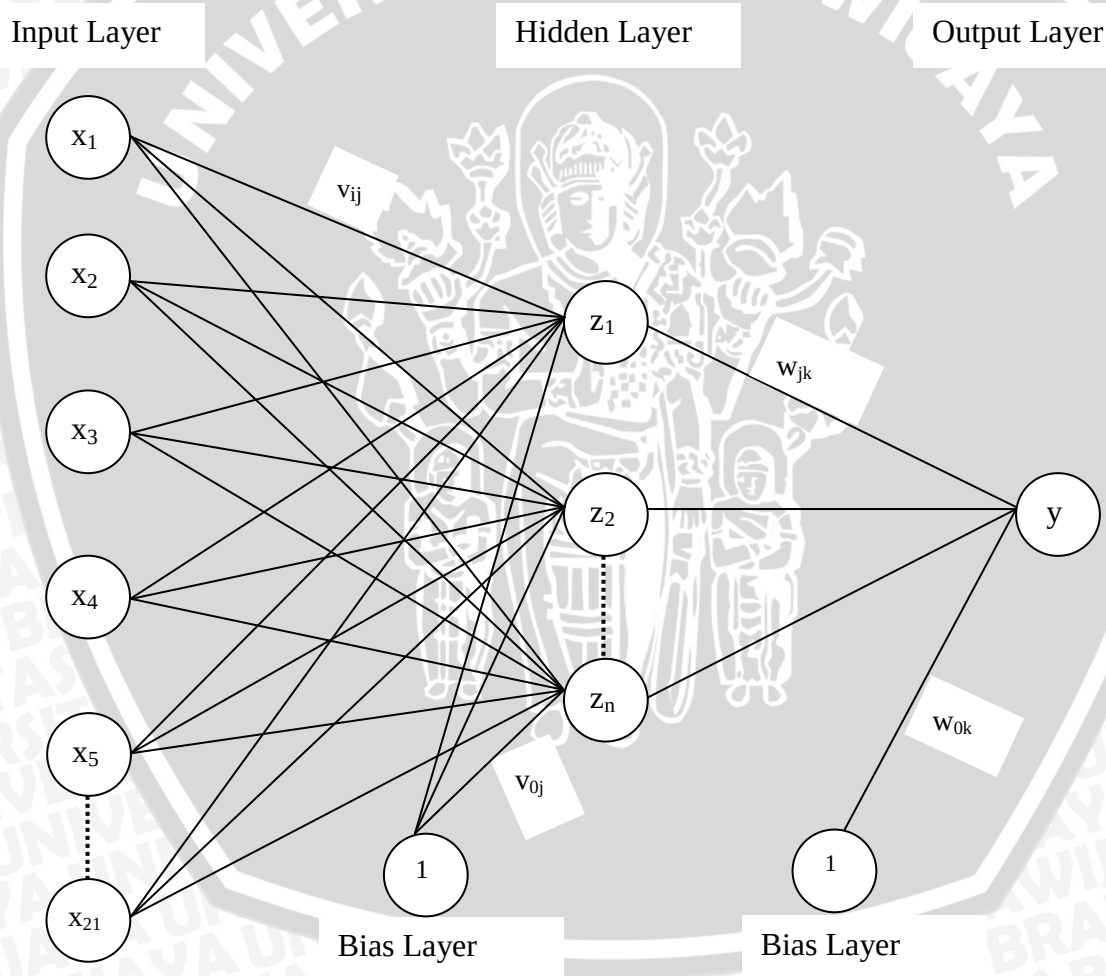
Hasil keluaran tersebut berupa sebuah neuron pada *output layer*. Hasil keluaran ini pada saat pengujian akan dicocokkan dengan nilai target pada data asli, sehingga akan diketahui tingkat akurasi sistem.

3.2.4.3 Arsitektur Jaringan

Arsitektur jaringan dari *resilient backpropagation* adalah menggunakan *multilayer perceptron* yang terdiri dari lapisan masukan *(input layer)*, lapisan tersembunyi *(hidden layer)* dan lapisan keluaran *(output layer).* Arsitektur yang digunakan adalah 21 unit *neuron* pada *input layer*, *n* unit *neuron* pada *hidden layer*, dan 1 unit *neuron* pada *output layer*. *Input layer* yang memiliki 21 unit

neuron ($x_1 - x_{21}$) terhubung langsung dengan *hidden layer* yang memiliki *n* unit *neuron* pada *hidden layer* (z). Hubungan neuron-neuron pada *input layer* dan *hidden layer* tersebut ditentukan oleh bobot *dari input layer* ke *hidden layer* (v_{ij}) dan bias ke *hidden layer* ($v_{0}j$). Kemudian, *n* unit *neuron* pada *hidden layer* terhubung langsung dengan *output layer* yang memiliki 1 unit *neuron* (*y*), yang besarnya ditentukan oleh bobot dari *hidden layer* ke *output layer* (w_{jk}) dan bias ke *output layer* (w_{0k}).

Gambar arsitektur jaringan algoritma *resilient backpropagation* untuk klasifikasi status tahapan keluarga sejahtera dapat dilihat pada Gambar 3.3.



Gambar 3. 3 Arsitektur JST *Resilient Backpropagation*

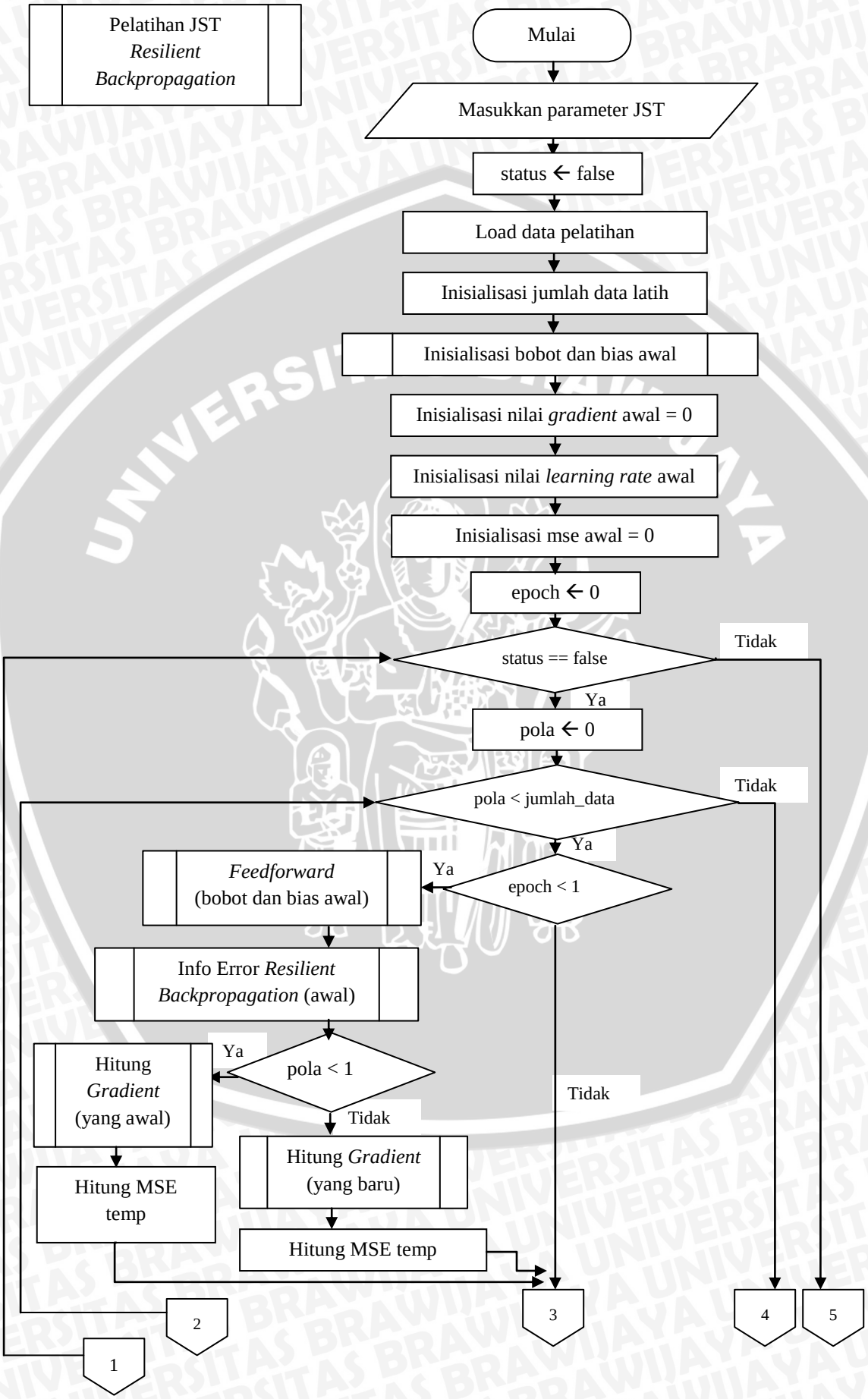
3.2.5 Perancangan Proses Pelatihan JST *Resilient Backpropagation*

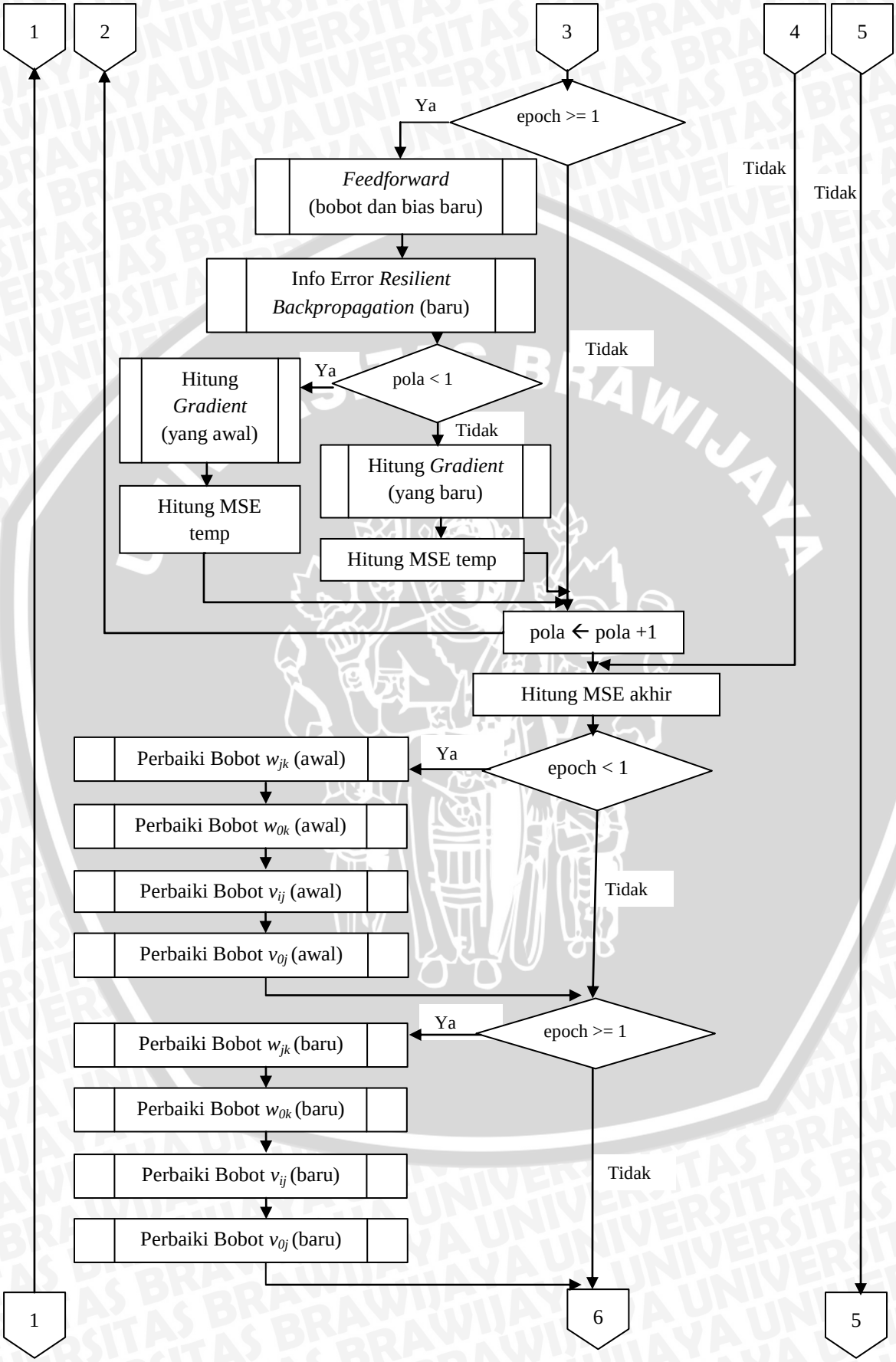
Perancangan proses pelatihan JST *resilient backpropagation* terdiri dari 5 perancangan proses agar JST dapat mengenali pola, yaitu perancangan proses inisialisasi bobot dan bias awal, perancangan proses *feedforward*, perancangan proses info *error resilient backpropagation*, perancangan proses hitung *gradient*, dan perancangan proses perbaikan bobot.

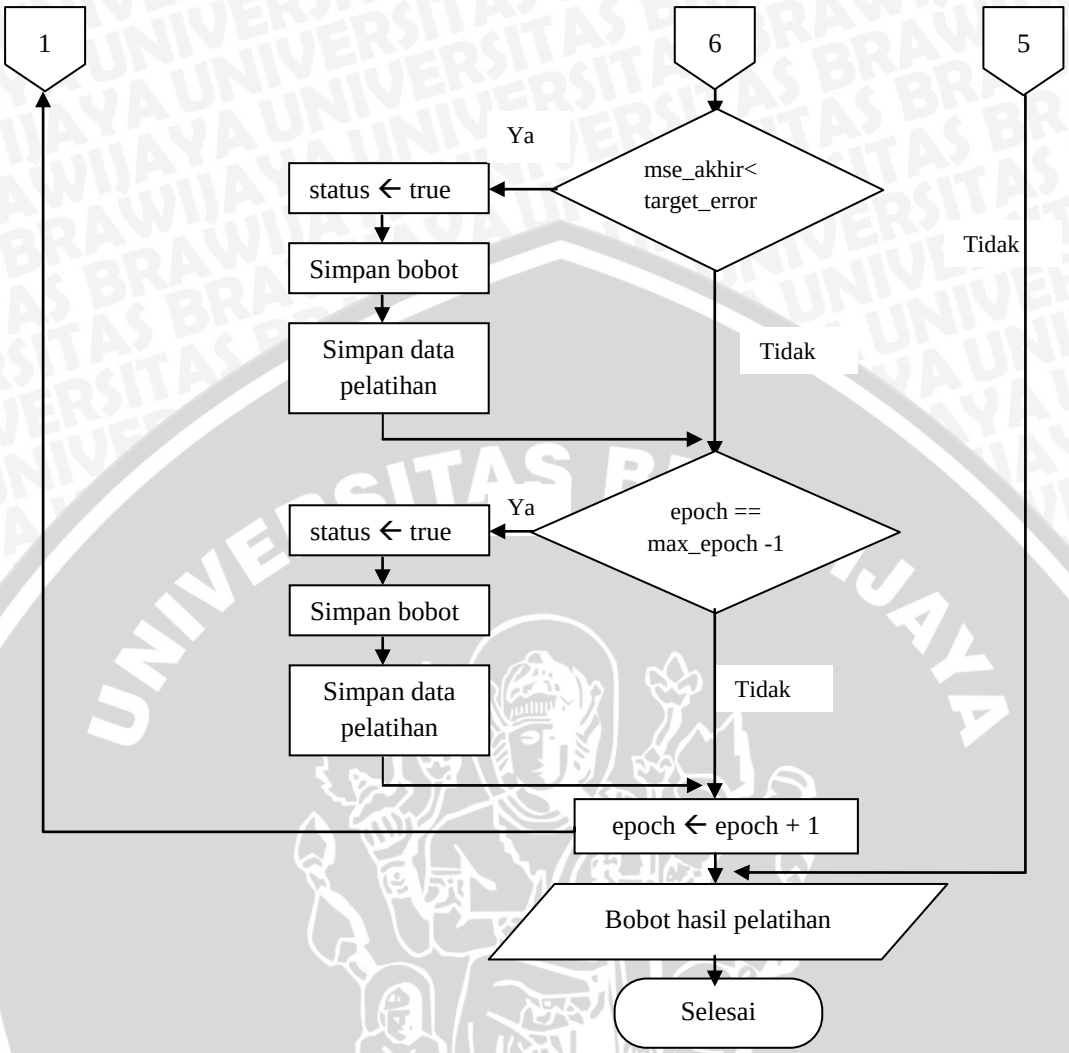
Secara keseluruhan, langkah-langkah pelatihan dengan menggunakan algoritma *resilient backpropagation* dapat digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.4 dan langkah-langkahnya sebagai berikut:

1. Mulai.
2. Masukkan nilai dari parameter JST, yaitu jumlah *neuron* pada *hidden layer*, *error target*, *learning rate*, faktor penaik (η^+), faktor penurun (η^-), penyesuaian minimum (Δ_{min}), penyesuaian maksimum (Δ_{max}) dan maksimum *epoch*.
3. Masukkan data pelatihan yang berupa data input (nilai dari 21 indikator status tahapan keluarga sejahtera).
4. Inisialisasi jumlah data latih keseluruhan.
5. Inisialisasi bobot-bobot dan bias awal seperti ditunjukkan pada Persamaan (2-5) sampai Persamaan (2-7).
6. Inisialisasi nilai *gradient* awal sama dengan 0.
7. Inisialisasi nilai *learning rate* awal.
8. Inisialisasi nilai MSE awal sama dengan 0.
9. Set *epoch* mula-mula sama dengan 0.
10. Setelah bobot dan bias sudah terkoneksi dengan jaringan, maka lakukan tahap *feedforward*, info *error resilient backpropagation*, hitung *gradient*, dan hitung nilai MSE. Lakukan tahapan-tahapan ini sampai pola data yang terakhir.
11. Lakukan perbaikan bobot dan bias setiap kali *epoch*.
12. Periksa apakah *epoch* sudah mencapai maksimum *epoch* atau kesalahan *output* (MSE) lebih kecil dari kesalahan yang ditargetkan (*error target*)? Jika ya, maka kerjakan langkah 14 dan seterusnya. Jika tidak, kerjakan langkah 13.
13. Naikkan *epoch* 1, ulangi pada langkah 10 dan seterusnya.

14. Proses pelatihan pada jaringan dihentikan, kemudian bobot dan bias akhir hasil dari pelatihan disimpan pada suatu dataset.
15. Simpan data pelatihan ke dalam tabel history_pelatihan.
16. Bobot dan bias akhir hasil pelatihan yang telah tersimpan sudah siap untuk digunakan pada proses pengujian.
17. Selesai.


Pelatihan JST *Resilient Backpropagation*
Mulai
Masukkan parameter JST
status ← false
Load data pelatihan
Inisialisasi jumlah data latih
Inisialisasi bobot dan bias awal
Inisialisasi nilai *gradient* awal = 0
Inisialisasi nilai *learning rate* awal
Inisialisasi mse awal = 0
epoch ← 0
status == false
Tidak
Ya
pola ← 0
pola < jumlah_data
Tidak
Ya
Feedforward (bobot dan bias awal)
Ya
epoch < 1
Info Error *Resilient Backpropagation* (awal)
Hitung *Gradient* (yang awal)
Ya
pola < 1
Tidak
Tidak
Hitung *Gradient* (yang baru)
Hitung MSE temp
Hitung MSE temp
1
2
3
4
5


1
2
3
4
5
epoch >= 1
Ya
Tidak
Tidak
Feedforward
(bobot dan bias baru)
Info Error Resilient Backpropagation (baru)
Tidak
pola < 1
Ya
Hitung Gradient (yang awal)
Tidak
Hitung Gradient (yang baru)
Hitung MSE temp
Hitung MSE temp
pola ← pola +1
Hitung MSE akhir
epoch < 1
Ya
Perbaiki Bobot w_{jk} (awal)
Perbaiki Bobot w_{0k} (awal)
Perbaiki Bobot v_{ij} (awal)
Perbaiki Bobot v_{0j} (awal)
Tidak
epoch >= 1
Ya
Perbaiki Bobot w_{jk} (baru)
Perbaiki Bobot w_{0k} (baru)
Perbaiki Bobot v_{ij} (baru)
Perbaiki Bobot v_{0j} (baru)
Tidak
1
6
5



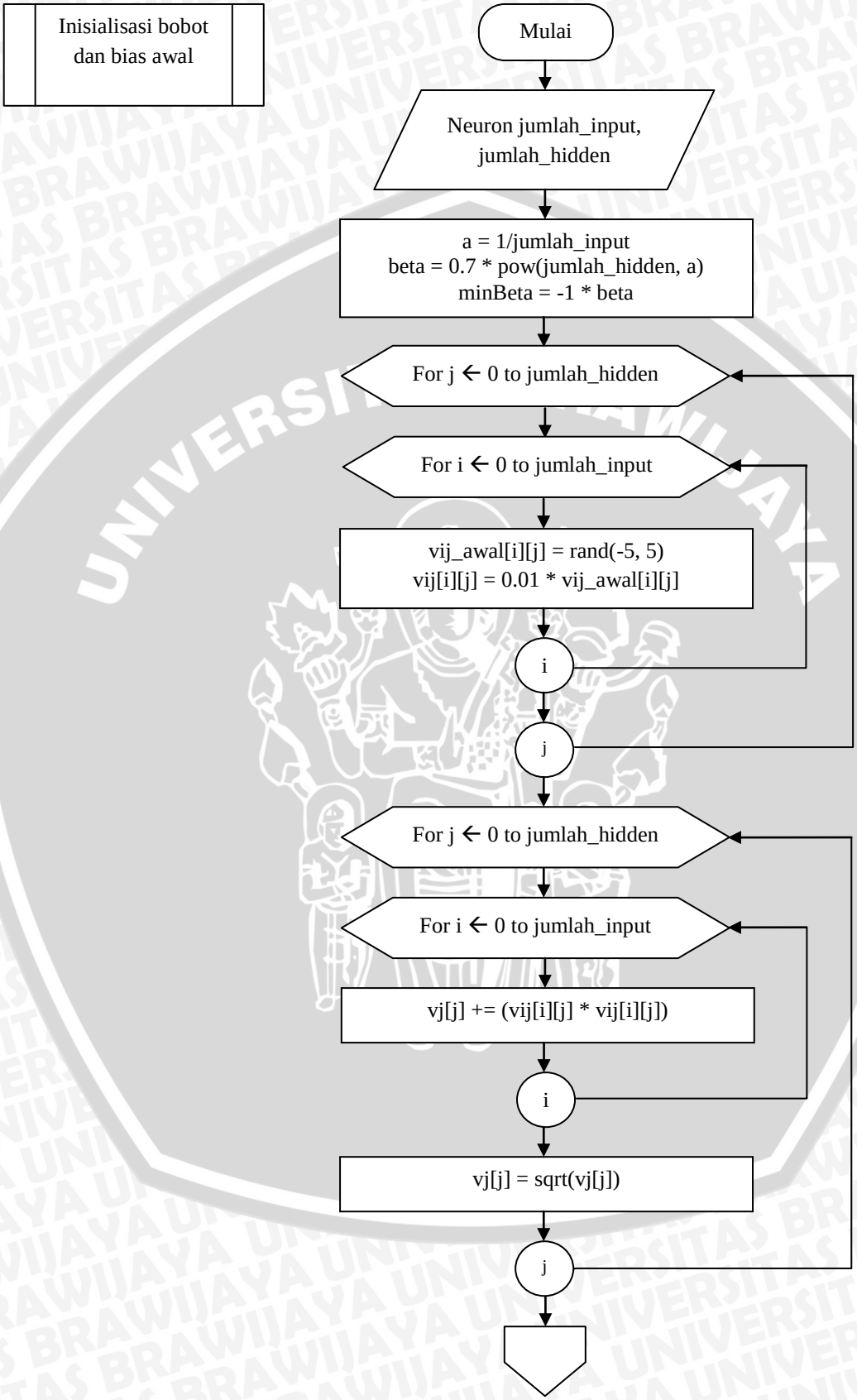
Gambar 3. 4 Diagram alir proses pelatihan JST *Resilient Backropagation*

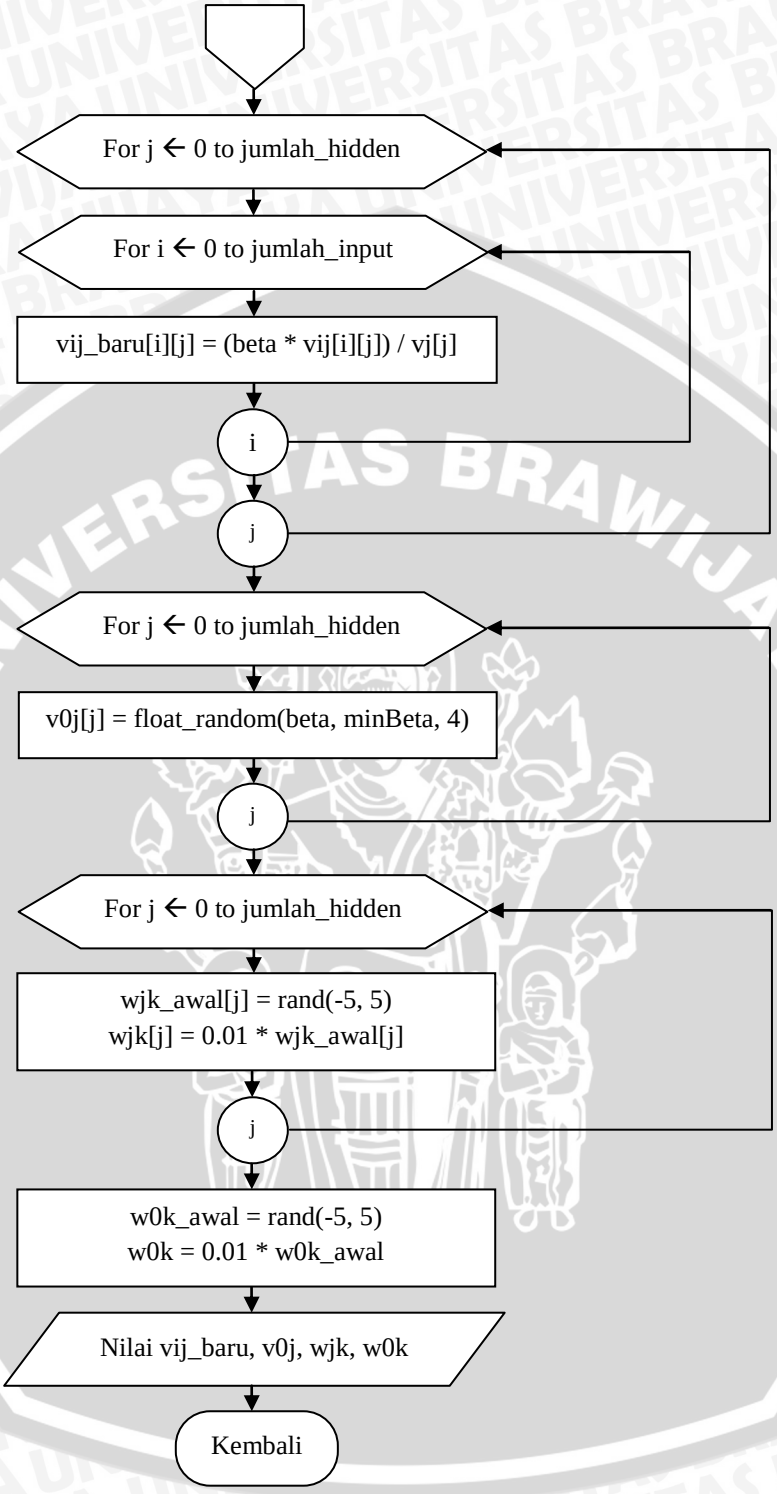
3.2.5.1 Perancangan Proses Inisialisasi Bobot dan Bias Awal

Inisialisasi bobot dan bias awal menggunakan algoritma Nguyen Widrow. Algoritma ini membuat inisialisasi bobot dan bias ke *hidden layer*, sehingga dapat menghasilkan iterasi lebih cepat. Adapun proses inisialisasi bobot dan bias awal dengan dapat digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.5 dan langkah-langkahnya adalah sebagai berikut :

1. Mulai.
2. Tentukan jumlah *neuron* pada *input layer* dan *neuron* pada *hidden layer*.
3. Hitung harga faktor penskalaan β sesuai dengan Persamaan (2-5).

4. Inisialisasi nilai awal pada semua bobot $v_{ij(lama)}$ dari *input layer* ke *hidden layer* dengan bilangan acak dalam interval -0.05 sampai 0.05.
5. Hitung nilai $\|v_j\|$ sesuai dengan Persamaan (2-6).
6. Selanjutnya, hitung nilai v_{ij}, yaitu bobot yang dipakai sebagai inisialisasi sesuai dengan Persamaan (2-7).
7. Lakukan inisialisasi nilai awal pada bias v_{0j} dari *input layer* ke *hidden layer* dengan bilangan acak dalam interval $-\beta$ sampai β.
8. Lakukan inisialisasi nilai awal pada semua bobot w_{jk} dari *hidden layer* ke *output layer* dengan bilangan acak dalam interval -0.05 sampai 0.05.
9. Lakukan inisialisasi nilai awal pada bias w_{0k} dari *hidden layer* ke *output layer* dengan bilangan acak dalam interval -0.05 sampai 0.05.
10. Keluaran yang dihasilkan adalah nilai bobot dari input layer ke hidden layer (v_{ij}), bias dari *input layer* ke *hidden layer* (v_{0j}), bobot dari *hidden layer* ke *output layer* (w_{jk}) dan bias dari *hidden layer* ke *output layer* (w_{0k}).
11. Kembali.


Inisialisasi bobot dan bias awal
Mulai
Neuron jumlah_input, jumlah_hidden
a = 1/jumlah_input
beta = 0.7 * pow(jumlah_hidden, a)
minBeta = -1 * beta
For j ← 0 to jumlah_hidden
For i ← 0 to jumlah_input
vij_awal[i][j] = rand(-5, 5)
vij[i][j] = 0.01 * vij_awal[i][j]
i
j
For j ← 0 to jumlah_hidden
For i ← 0 to jumlah_input
vj[j] += (vij[i][j] * vij[i][j])
i
vj[j] = sqrt(vj[j])
j

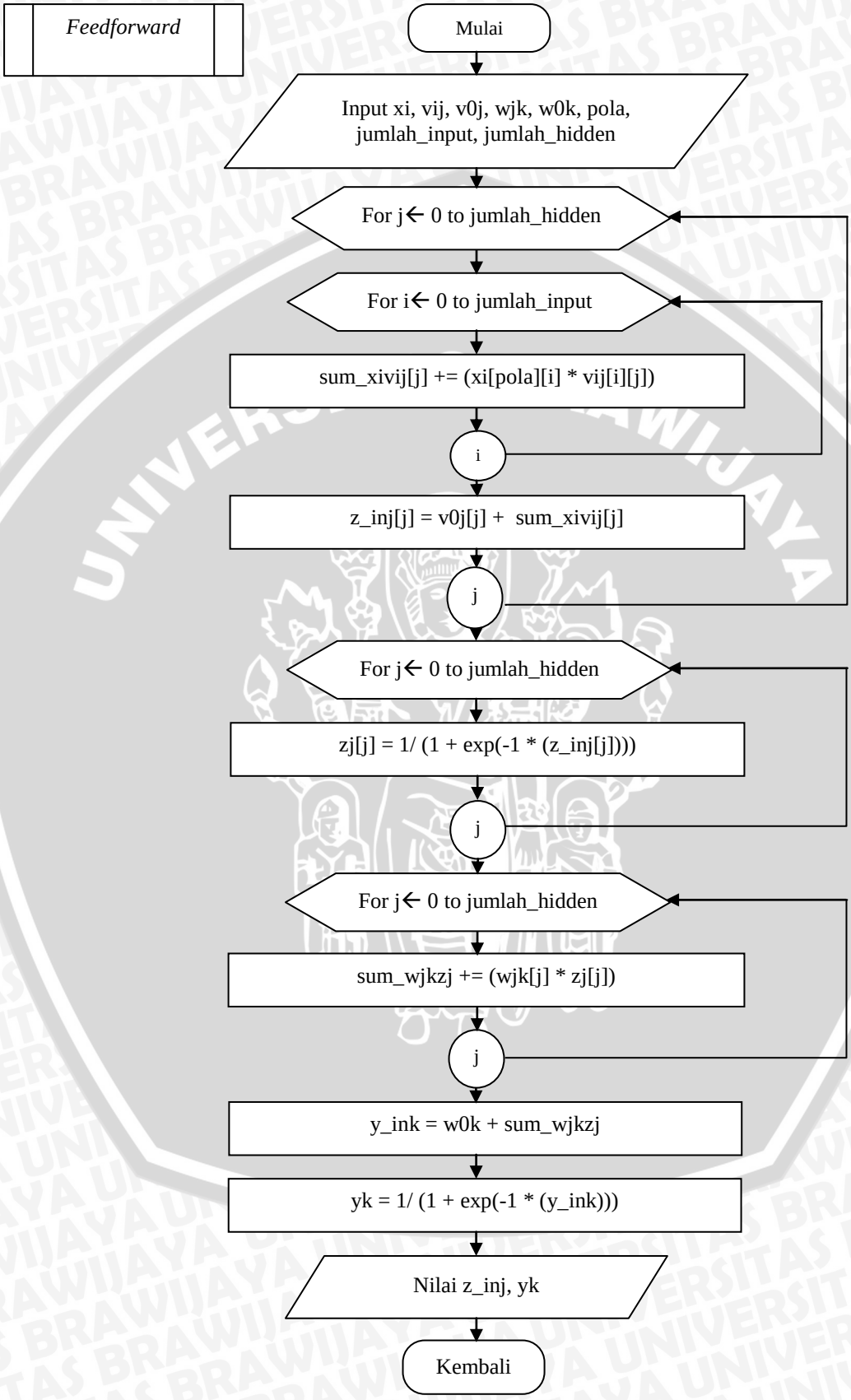


Gambar 3. 5 Diagram alir proses inisialisasi bobot dan bias awal

3.2.5.2 Perancangan Proses *Feedforward*

Proses perambatan maju *(feedforward)* dapat digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.6 dan langkah-langkahnya adalah sebagai berikut :

1. Mulai.
2. Jumlahkan sinyal-sinyal input terbobot pada tiap-tiap unit di *hidden layer*, yaitu dengan mengalikan setiap input x_i dengan bobot v_{ij} yang terhubung antara *input layer* dengan *hidden layer*. Kemudian, Hitung sinyal-sinyal masukan pada *hidden layer* z_in_j dengan cara menjumlahkan seluruh vektor bobot yang menuju *neuron hidden* yang sama dengan menggunakan Persamaan (2-13).
3. Hitung sinyal keluarannya dari z_in_j dengan fungsi aktivasi sigmoid biner seperti pada Persamaan (2-14).
4. Jumlahkan sinyal-sinyal input terbobot pada tiap-tiap unit di *output layer*, yaitu dengan mengalikan setiap hasil aktivasi (z_j) dengan bobot w_{jk} yang terhubung antara *hidden layer* dengan *output layer*. Kemudian, Hitung sinyal-sinyal masukan pada *output layer* y_in_k dengan cara menjumlahkan seluruh vektor bobot yang menuju neuron pada *output layer* dengan menggunakan Persamaan (2-15).
5. Hitung sinyal keluarannya dari z_in_j dengan fungsi aktivasi sigmoid biner dengan menggunakan Persamaan (2-16).
6. *Output* yang dihasilkan berupa nilai z_in_j, z_j, y_in_k, y_k.
7. Kembali.

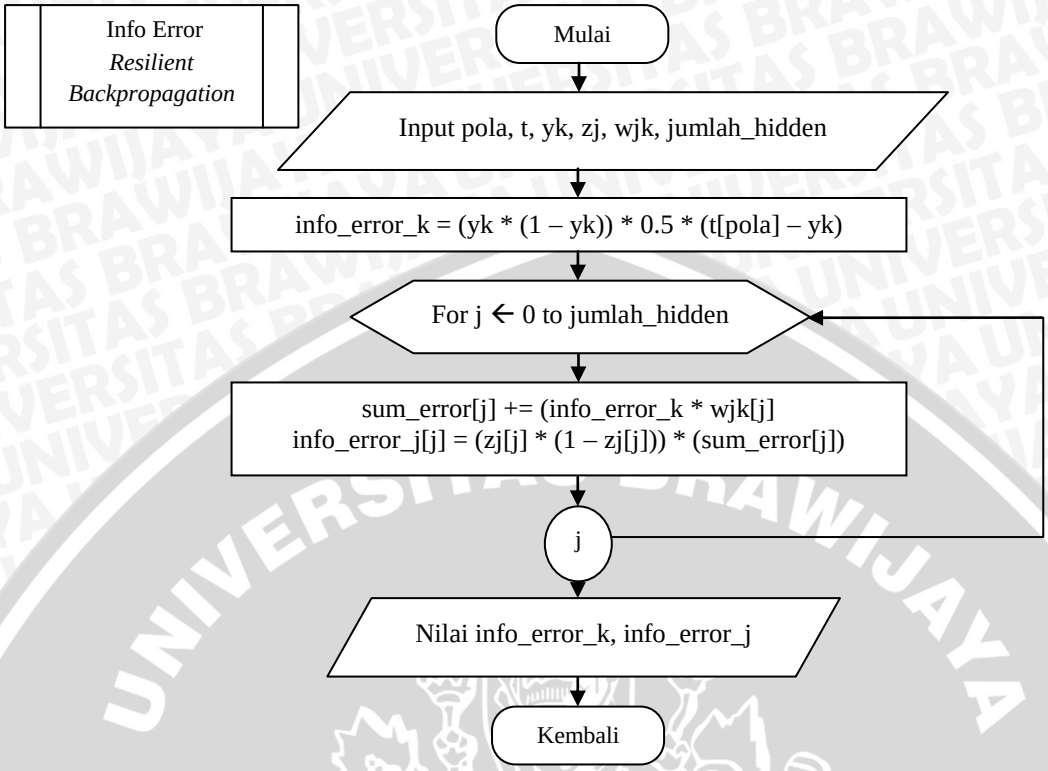


Gambar 3. 6 Diagram alir proses *feedforward*

3.2.5.3 Perancangan Proses Info *Error Resilient Backpropagation*

Proses info error *resilient backpropagation* merupakan proses perhitungan informasi kesalahan pada tiap *neuron* pada *output layer* hingga *hidden layer*. Adapun proses info *error resilient backpropagation* dapat digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.7 dan langkah-langkahnya adalah sebagai berikut :

1. Mulai.
2. Pada *output layer*, hitung selisih antara target pengenalan dengan output pengenalan, dan dikalikan 0.5. Hitung nilai informasi *error* δ_k pada tiap-tiap unit *output*, dengan cara mengalikan selisih tersebut dengan *output* yang telah diaktivasi dengan fungsi turunan aktivasi pada Persamaan (2-17).
3. Pada *hidden layer*, hitung nilai informasi *error* δ_j pada tiap-tiap unit di *hidden layer* dengan melakukan proses *backpropagation* terhadap *error* pada *output* δ_j, yaitu masing-masing nilai informasi *error* pada unit output dikalikan dengan bobot lama yang terhubung dengan neuron pada *output layer*. Kemudian hasil perkalian pada seluruh koneksi yang terhubung dengan masing-masing neuron pada *hidden layer*. Perhitungan ini menggunakan Persamaan (2-18).
4. *Output* yang dihasilkan adalah nilai informasi *error* j (δ_j), informasi *error* k (δ_k).
5. Kembali.

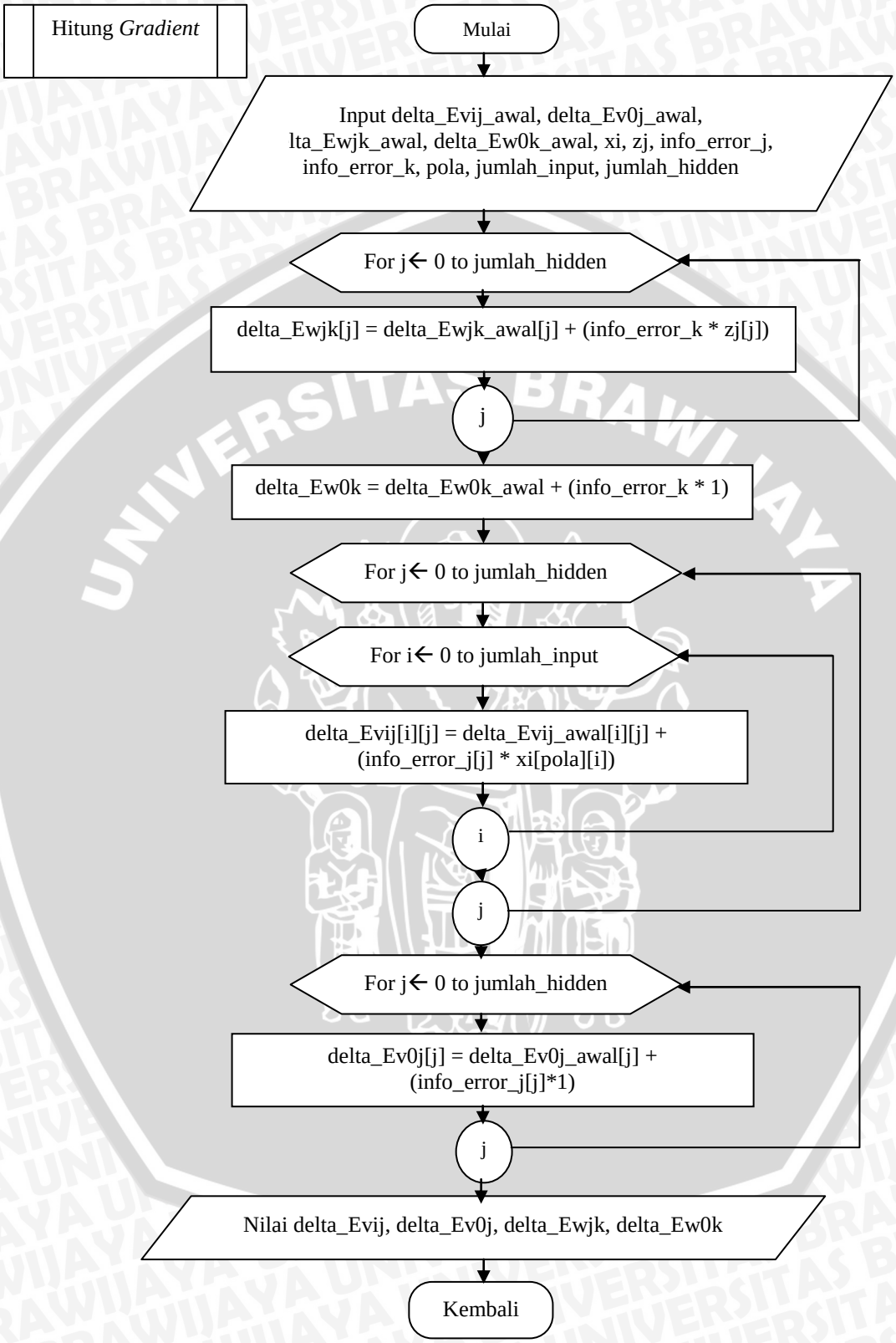


Gambar 3. 7 Diagram alir proses hitung info *error Resilient Backpropagation*

3.2.5.4 Perancangan Proses Hitung *Gradient*

Proses perhitungan *gradient* dilakukan untuk mengetahui arah perubahan *gradient*. Proses perhitungan *gradient* ditunjukkan pada Gambar 3.8.

1. Mulai.
2. Menghitung nilai *gradient*, yaitu dengan cara menghitung nilai turuanan pertama fungsi *error* terhadap bobot pada *hidden layer* ke *output layer* dengan menggunakan Persamaan (2-20).
3. Selanjutnya, menghitung nilai turuanan pertama fungsi *error* terhadap bobot pada *input* ke *hidden layer* dengan menggunakan Persamaan (2-19).
4. *Output* yang dihasilkan adalah nilai turunan fungsi *error* terhadap bobot pada *hidden layer* ke *output layer* $\left(\frac{\partial E}{\partial w_{jk}}^{(t)}\right)$, turunan fungsi *error* terhadap bobot pada *input* ke *hidden layer* $\left(\frac{\partial E}{\partial v_{ij}}^{(t)}\right)$.
5. Kembali.



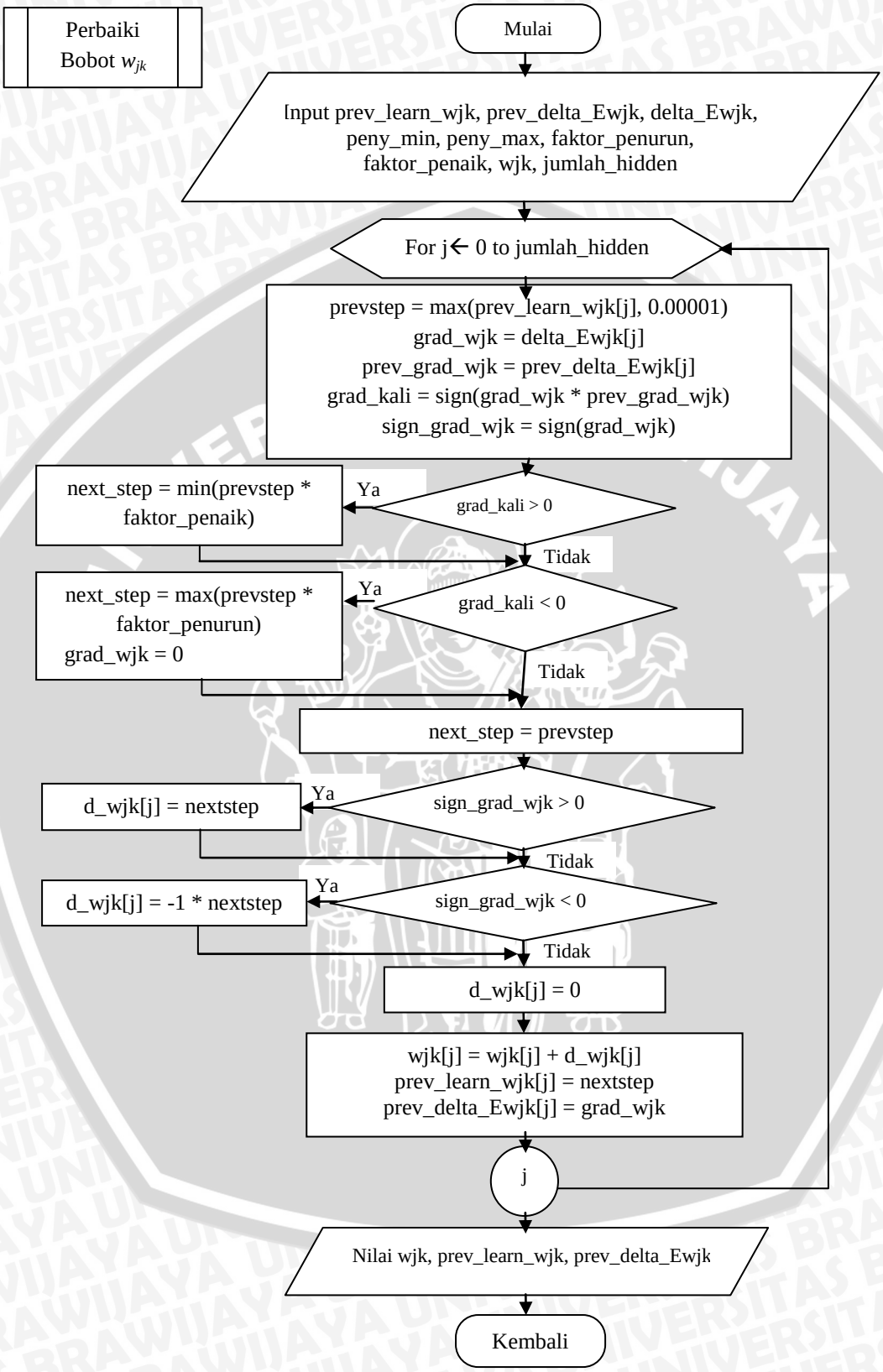
Gambar 3. 8 Diagram alir proses hitung *gradient*

3.2.5.5 Perancangan Proses Perbaikan Bobot

Proses perbaikan bobot dengan algoritma *resilient backpropagation* ada empat macam bobot yang harus diperbaiki, yaitu bobot antara *hidden layer* ke *output layer* (w_{jk}), bobot antara bias ke *output layer* (w_{0k}), bobot antara *input layer* ke *hidden layer* (v_{ij}), dan bobot antara bias ke *hidden layer* (v_{0j}).

Proses perbaikan bobot antara *hidden layer* ke *output layer* (w_{jk}) digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.9 dan langkah-langkahnya adalah sebagai berikut :

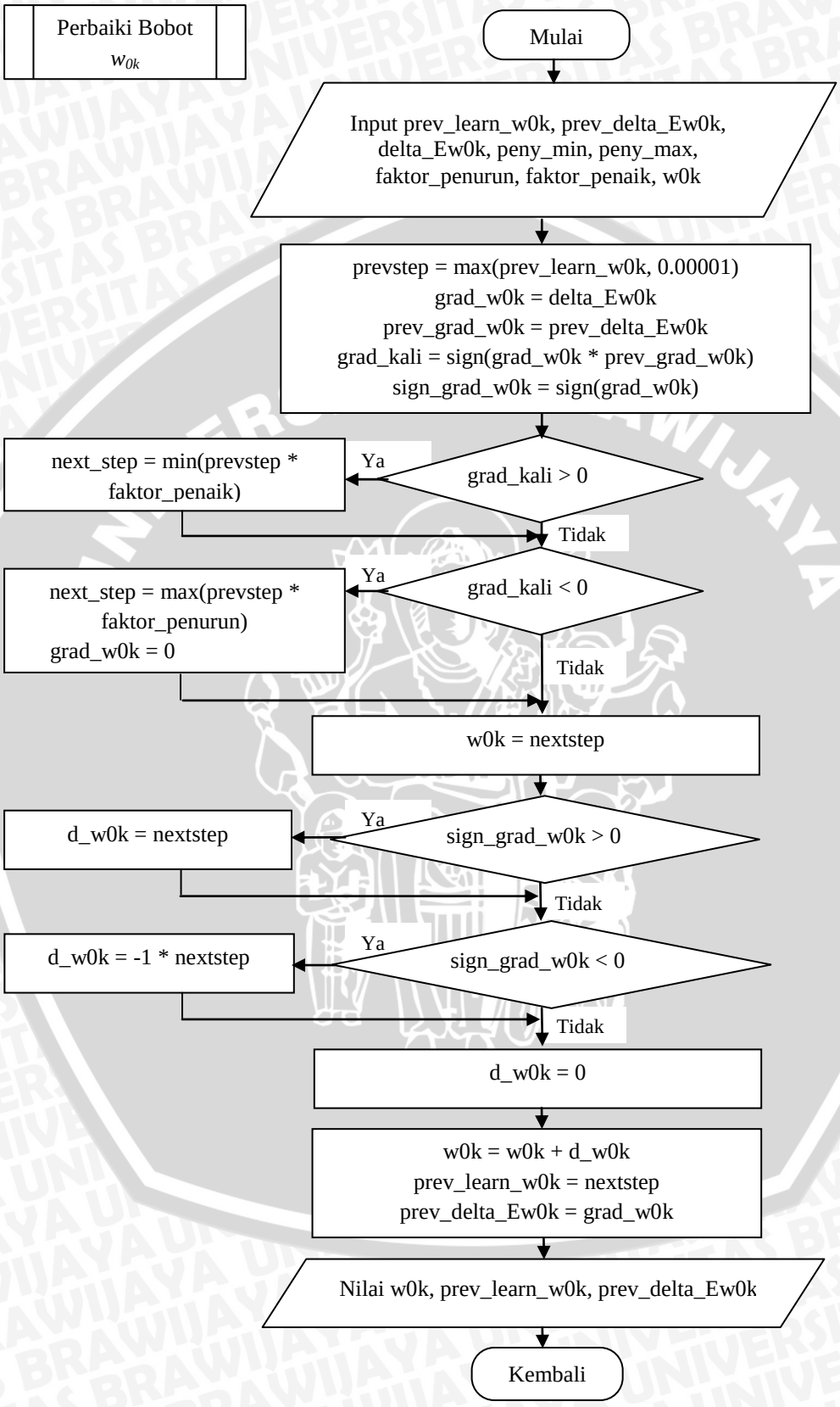
1. Mulai.
2. Menghitung nilai *learning rate* yang baru pada bobot antara *hidden layer* ke *output layer* (Δ_{jk}) dengan cara menghitung nilai *gradient* lama dikalikan dengan *gradient* saat itu, aturan perbaikan *learning rate* dapat dilihat pada Persamaan (2-21).
3. Selanjutnya, melakukan perbaikan nilai bobot antara *hidden layer* ke *output layer* (w_{jk}), dengan ketentuan sesuai pada Persamaan (2-22) dan Persamaan (2-23).
4. *Output* yang dihasilkan adalah nilai bobot baru antara *hidden layer* ke *output layer* ($w_{jk}^{(t+1)}$), nilai *learning rate* baru ($\Delta_{ij}^{(t)}$), dan nilai *gradient* baru ($\frac{\partial E}{\partial w_{ij}}^{(t)}$).
5. Kembali.



Gambar 3. 9 Diagram alir proses perbaikan bobot w_{jk}

Proses perbaikan bobot antara bias ke *output layer* (w_{0k}) digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.10 dan langkah-langkahnya adalah sebagai berikut :

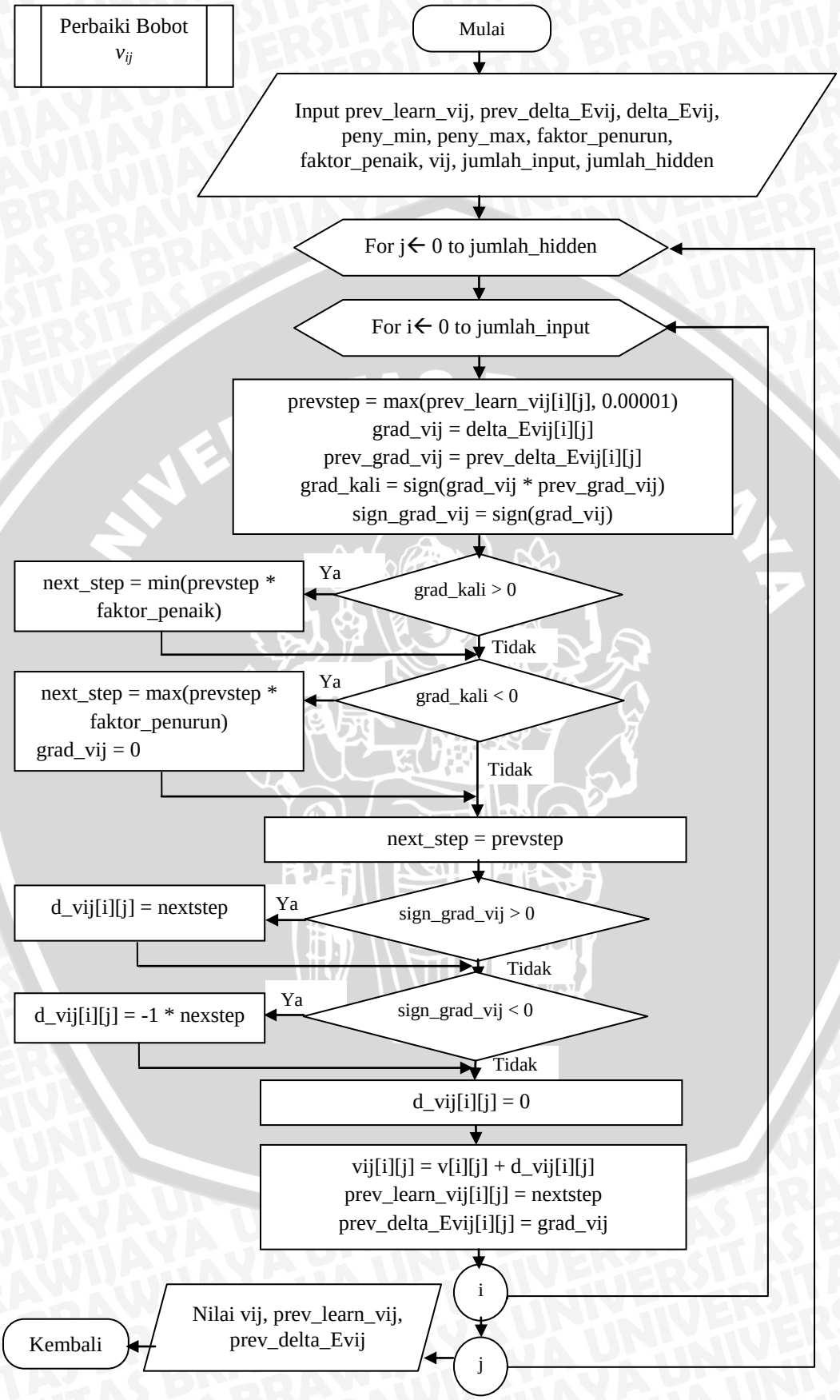
1. Mulai.
2. Menghitung nilai *learning rate* yang baru pada bobot antara *bias* ke *output layer* (Δ_{0k}) dengan cara menghitung nilai *gradient* lama dikalikan dengan *gradient* saat itu, aturan ini dapat dilihat pada Persamaan (2-21).
3. Selanjutnya, melakukan perbaikan nilai bobot antara *bias* ke *output layer* (w_{0k}), dengan ketentuan sesuai pada Persamaan (2-22) dan Persamaan (2-23).
4. *Output* yang dihasilkan adalah nilai bobot baru antara bias ke *output layer* ($w_{0k}^{(t+1)}$), nilai *learning rate* baru ($\Delta_{0k}^{(t)}$), dan nilai *gradient* baru ($\frac{\partial E}{\partial w_{0k}}^{(t)}$)
5. Kembali.



Gambar 3. 10 Diagram alir proses perbaikan bobot w_{0k}

Proses perbaikan bobot antara *input layer* ke *hidden layer* (v_{ij}) digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.11 dan langkah-langkahnya adalah sebagai berikut :

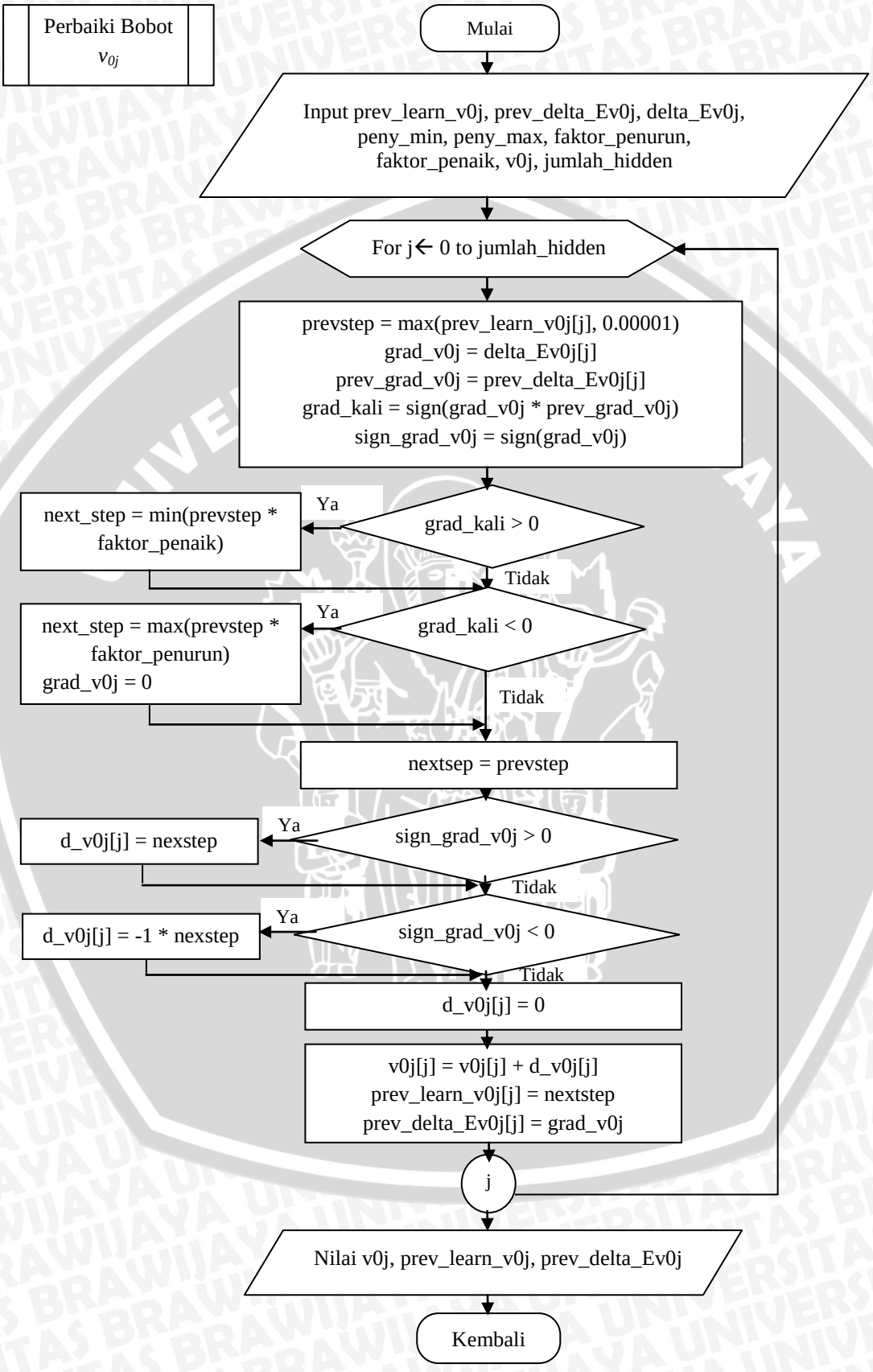
1. Mulai.
2. Menghitung nilai *learning rate* yang baru pada bobot antara *input layer* ke *hidden layer* (Δ_{ij}) dengan cara menghitung nilai *gradient* lama dikalikan dengan *gradient* saat itu, aturan ini dapat dilihat pada Persamaan (2-21).
3. Selanjutnya, melakukan perbaikan nilai bobot antara *input layer* ke *hidden layer* (v_{ij}), dengan ketentuan sesuai pada Persamaan (2-22) dan Persamaan (2-23).
4. *Output* yang dihasilkan adalah nilai bobot baru antara *input layer* ke *hidden layer* ($v_{ij}^{(t+1)}$), nilai *learning rate* baru ($\Delta_{ij}^{(t)}$), dan nilai *gradient* baru ($\frac{\partial E}{\partial v_{ij}}^{(t)}$).
5. Kembali.



Gambar 3. 11 Diagram alir proses perbaikan bobot v_{ij}

Proses perbaikan bobot antara bias ke *hidden layer* (v_{0j}) digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.12 dan langkah-langkahnya adalah sebagai berikut :

1. Mulai.
2. Menghitung nilai *learning rate* yang baru pada bobot antara bias ke *hidden layer* (Δ_{0j}) dengan cara menghitung nilai *gradient* lama dikalikan dengan *gradient* saat itu, aturan ini dapat dilihat pada Persamaan (2-21) seperti halnya pada perbaikan bobot antara *input layer* ke *hidden layer*.
3. Selanjutnya, melakukan perbaikan nilai bobot antara bias ke *hidden layer* (v_{0j}), dengan ketentuan sesuai pada Persamaan (2-22) dan Persamaan (2-23) seperti halnya pada perbaikan bobot antara *input layer* ke *hidden layer*.
4. *Output* yang dihasilkan adalah nilai bobot baru antara bias ke *hidden layer* ($v_{0j}^{(t+1)}$), nilai *learning rate* baru ($\Delta_{0j}^{(t)}$), dan nilai *gradient* baru ($\frac{\partial E}{\partial v_{0j}}^{(t)}$).
5. Kembali.

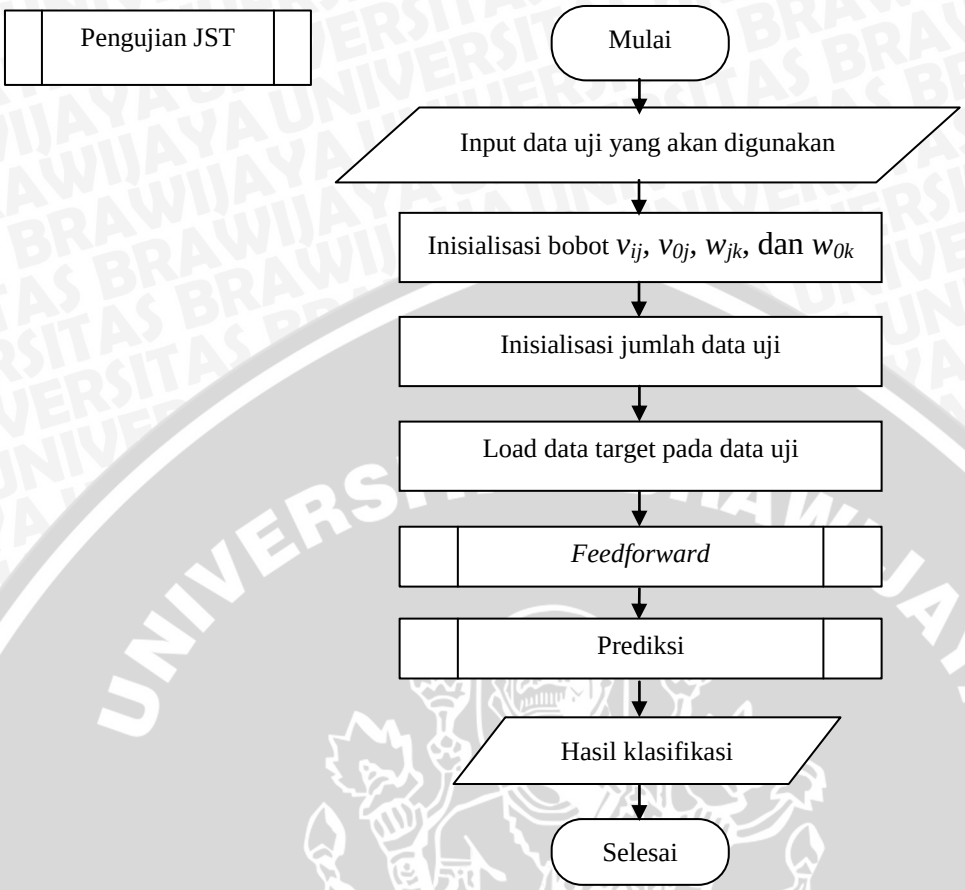


Gambar 3. 12 Diagram alir proses perbaikan bobot w_{0k}

3.2.6 Perancangan Proses Pengujian JST

Tahap pengujian JST *resilient backpropagation* diterapkan dengan menggunakan algoritma *feedforward* seperti pada tahap pelatihan. Adapun proses pengujian secara keseluruhan *JST resilient backpropagation* dapat digambarkan dalam bentuk diagram alir seperti yang terlihat pada Gambar 3.13 dan langkah-langkahnya adalah sebagai berikut :

1. Mulai.
2. Input atau load data uji ke yang akan digunakan.
3. Inisialisasi bobot v_{ij}, v_{0j}, w_{jk}, dan w_{0k} dari hasil pelatihan yang telah tersimpan.
4. Inisialisasi jumlah data uji.
5. Load data target pada data uji yang akan digunakan.
6. Lakukan *feedforward* yang langkah-langkahnya ditunjukkan seperti pada Gambar 3.6. Setelah proses *feedforward* selesai, maka output yang dihasilkan tidak akan diproses lagi menuju *resilient backpropagation*.
7. Lakukan prediksi pada klasifikasi status tahapan keluarga sejahtera.
8. Output berupa salah satu hasil klasifikasi berdasarkan 5 kelas yang ada, yaitu tahap keluarga prasejahtera, keluarga sejahtera I, keluarga sejahtera II, keluarga sejahtera III, atau keluarga sejahtera III plus.
9. Selesai.

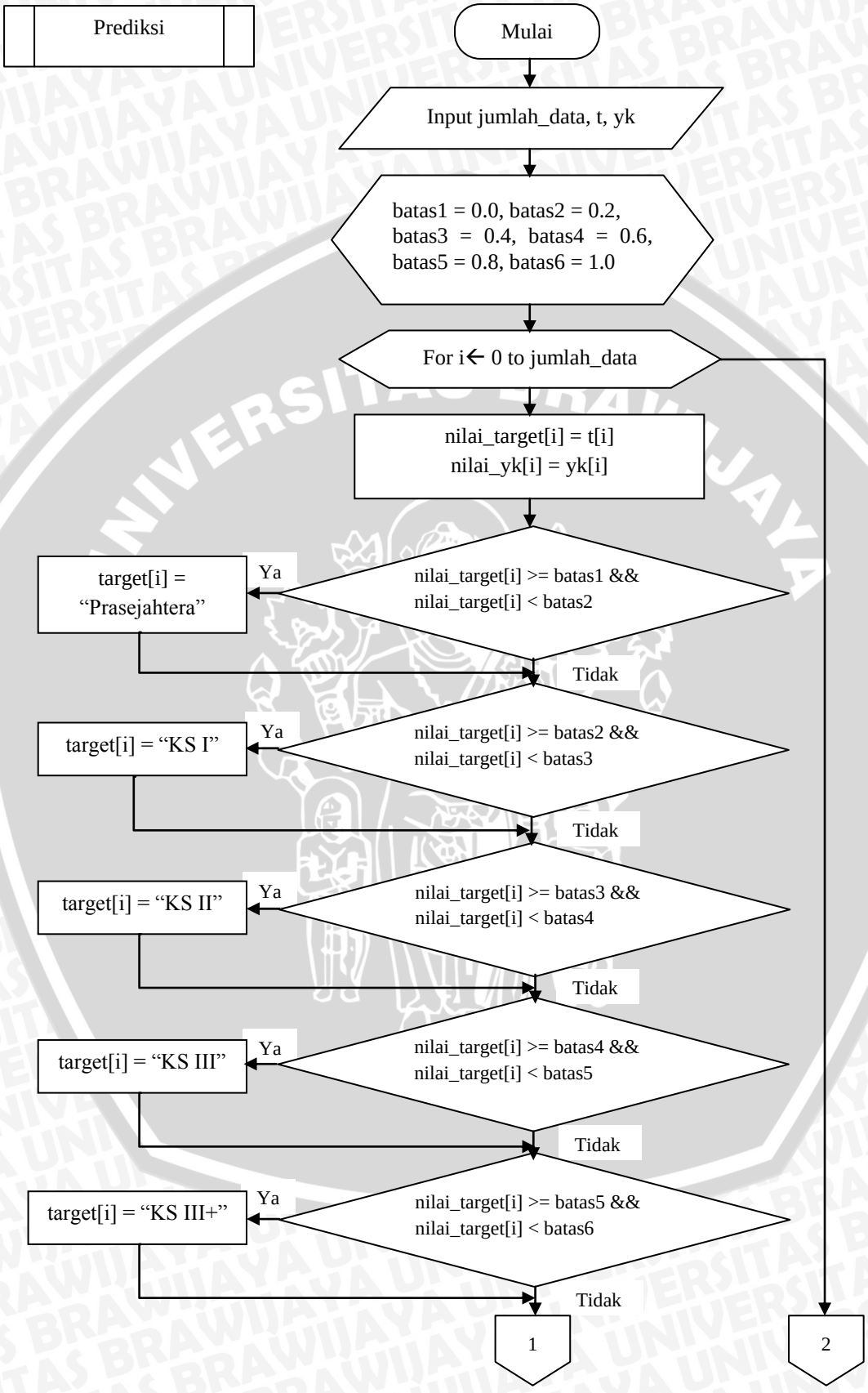


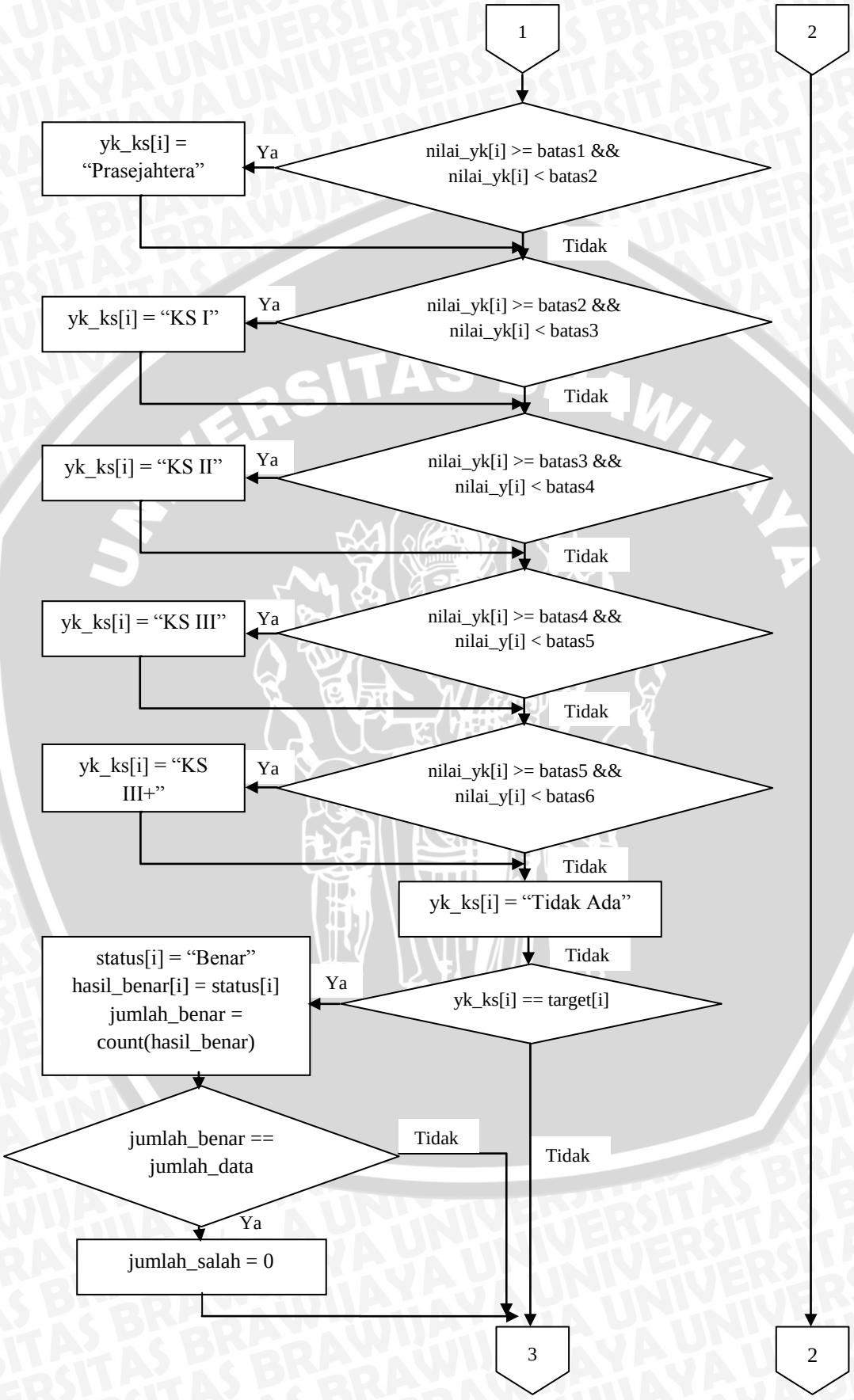
Gambar 3. 13 Diagram alir proses pengujian JST

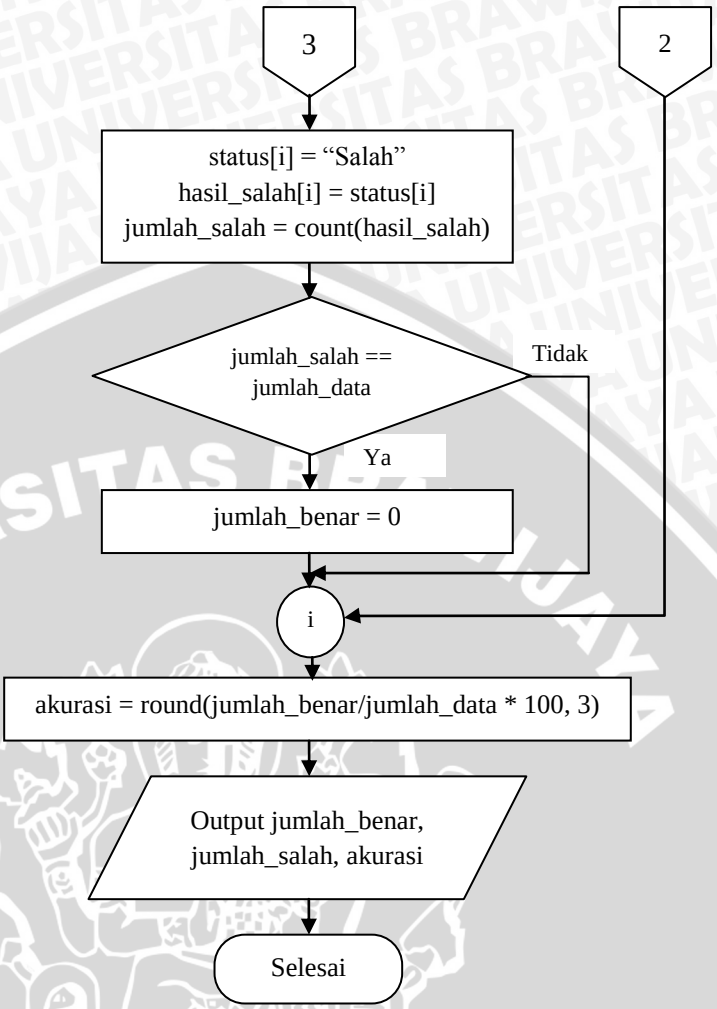
3.2.7 Perancangan Proses Prediksi

Proses prediksi klasifikasi status tahapan keluarga sejahtera ditunjukkan pada Gambar 3.14 dan langkah-langkahnya sebagai berikut :

1. Mulai.
2. Input jumlah data, target, dan nilai y_k.
3. Inisialisasi batas1 sampai batas6.
4. Lakukan seleksi kondisi nilai target pada data asli dengan batas1 sampai batas6.
5. Lakukan seleksi kondisi nilai y_k hasil *feedforward* pada pengujian batas1 sampai batas6.
6. Hitung hasil prediksi sistem dengan target data asli, hitung jumlah data benar dan jumlah data salah.
7. Output berupa jumlah benar, jumlah salah, dan akurasi prediksi.
8. Selesai.


Prediksi
Mulai
Input jumlah_data, t, yk
batas1 = 0.0, batas2 = 0.2,
batas3 = 0.4, batas4 = 0.6,
batas5 = 0.8, batas6 = 1.0
For i← 0 to jumlah_data
nilai_target[i] = t[i]
nilai_yk[i] = yk[i]
nilai_target[i] >= batas1 &&
nilai_target[i] < batas2
Ya
target[i] =
“Prasejahtera”
Tidak
nilai_target[i] >= batas2 &&
nilai_target[i] < batas3
Ya
target[i] = “KS I”
Tidak
nilai_target[i] >= batas3 &&
nilai_target[i] < batas4
Ya
target[i] = “KS II”
Tidak
nilai_target[i] >= batas4 &&
nilai_target[i] < batas5
Ya
target[i] = “KS III”
Tidak
nilai_target[i] >= batas5 &&
nilai_target[i] < batas6
Ya
target[i] = “KS III+”
Tidak
1
2


1
2
nilai_yk[i] >= batas1 && nilai_yk[i] < batas2
Ya
yk_ks[i] = "Prasejahtera"
Tidak
nilai_yk[i] >= batas2 && nilai_yk[i] < batas3
Ya
yk_ks[i] = "KS I"
Tidak
nilai_yk[i] >= batas3 && nilai_y[i] < batas4
Ya
yk_ks[i] = "KS II"
Tidak
nilai_yk[i] >= batas4 && nilai_y[i] < batas5
Ya
yk_ks[i] = "KS III"
Tidak
nilai_yk[i] >= batas5 && nilai_y[i] < batas6
Ya
yk_ks[i] = "KS III+"
Tidak
yk_ks[i] = "Tidak Ada"
Tidak
yk_ks[i] == target[i]
Ya
status[i] = "Benar"
hasil_benar[i] = status[i]
jumlah_benar = count(hasil_benar)
jumlah_benar == jumlah_data
Tidak
Tidak
Ya
jumlah_salah = 0
3
2



Gambar 3. 14 Diagram alir proses prediksi

3.2.8 Contoh Perhitungan Manual

Diketahui 4 data sampel penelitian berupa indikator status tahapan keluarga sejahtera yang ditunjukkan pada Tabel 3.3 sampai Tabel 3.5.

Tabel 3. 3 Tabel data masukan indikator-1 sampai indikator-11

Data ke-	IND-1	IND-2	IND-3	IND-4	IND-5	IND-6	IND-7	IND-8	IND-9	IND-10	IND-11
1	v	v	v	v	v	v	v	v	v	v	v
2	v	v	x	v	v	v	v	v	v	v	v
3	v	v	v	v	-	-	v	v	v	v	v
4	v	v	v	v	-	v	-	v	v	v	v

Tabel 3. 4 Tabel data masukan indikator-12 sampai indikator-21

Data ke-	IND-12	IND-13	IND-14	IND-15	IND-16	IND-17	IND-18	IND-19	IND-20	IND-21
1	v	v	v	v	v	v	v	v	-	-
2	v	v	v	v	v	v	v	v	-	-
3	v	v	-	v	v	v	v	v	-	-
4	v	v	-	v	v	v	v	v	-	-

Tabel 3. 5 Tabel status data masukan

Data ke-	STATUS
1	KS III
2	KS I
3	KS II
4	KS III

Langkah pertama, yaitu mengidentifikasikan data masukan di atas sebagai x1 adalah indikator 1, x2 adalah indikator 2, sampai x21 adalah indikator 21. Kemudian dilakukan konversi data masukan sesuai dengan aturan pada Tabel 3.1 dan konversi data target dengan aturan pada Tabel 3.2. Apabila kolom indikator bertuliskan tanda "v", maka nilainya adalah 0.9, tanda "–" bernilai 0.7, tanda "x" bernilai 0.5, dan tanda "x*" bernilai 0.3. Nilai hasil konversi data masukan dan target ditunjukkan pada Tabel 3.6 sampai Tabel 3.8.

Tabel 3. 6 Tabel hasil konversi data masukan x_1 sampai x_{11}

Data ke-	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
2	0.9	0.9	0.5	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
3	0.9	0.9	0.9	0.9	0.7	0.7	0.9	0.9	0.9	0.9	0.9
4	0.9	0.9	0.9	0.9	0.7	0.9	0.7	0.9	0.9	0.9	0.9

Tabel 3. 7 Tabel hasil konversi data masukan x_{12} sampai x_{21}

Data ke-	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	x_{19}	x_{20}	x_{21}	Target
1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.7	0.7	0.7
2	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.7	0.7	0.3
3	0.9	0.9	0.7	0.9	0.9	0.9	0.9	0.9	0.7	0.7	0.5
4	0.9	0.9	0.7	0.9	0.9	0.9	0.9	0.9	0.7	0.7	0.7

Tabel 3. 8 Tabel hasil konversi target

Data ke-	Target
1	0.7
2	0.3
3	0.5
4	0.7

Parameter awal untuk pelatihan JST ditentukan sebagai berikut :

1. Jumlah unit input = 21
2. Jumlah unit hidden = 15

   Penentuan jumlah neuron pada *hidden layer* menggunakan Persamaan (2-8).

   Diketahui : $neuron_{input} = 21$

   $neuron_{output} = 1$

   Sehingga, jumlah neuron hidden adalah :

   $$neuron_{hidden} = (\frac{2}{3} \times 21) + 1 = 14 + 1 = 15$$

   Namun, untuk keperluan pengujian JST nantinya penentuan jumlah neuron pada *hidden layer* akan menggunakan metode percobaan *(trial and error).*
3. Jumlah unit output = 1
4. $\eta^- = 0.5$ dan $\eta^+ = 1.2$
5. $\Delta_0 = 0.1$
6. $\Delta_{min} = 0.000001 = 10^{-6}$ dan $\Delta_{max} = 50$
7. $\frac{\partial E(lama)}{\partial v_{ij}} = 0$ dan $\frac{\partial E(lama)}{\partial w_{jk}} = 0$

Proses Inisialisasi Bobot dan Bias Awal dengan Nguyen-Widrow

Inisialisasi bobot dan bias awal dengan menggunakan algoritma Nguyen-Widrow. Mula-mula dilakukan perhitungan harga faktor penskalaan β dengan Persamaan (2-5), sehingga nilai faktor penskalaan β diperoleh :

$$\beta = 0.7(\sqrt[21]{15}) = 0.7963$$

Selanjutnya, inisialisasi bobot-bobot dari *input layer* ke *hidden layer* mula-mula diberi nilai acak antara -0,05 hingga 0,05, sehingga diperoleh nilai inisialisasi bobot yang ditunjukkan pada Tabel 3.9.

Tabel 3. 9 Tabel hasil inisialisasi bobot v_{ij} dari *input* ke *hidden layer*

v_{ij}		I						
		x_1	x_2	x_3	x_4	$x_{...}$	$x_{...}$	x_{21}
J	Z_1	-0.04	-0.01	0.01	0.02	…	…	0.02
	Z_2	0.02	-0.03	0.01	-0.01	…	…	-0.02
	Z_3	-0.03	-0.03	0.04	0.03	…	…	0.01
	Z_4	0.04	-0.02	0.01	0.01	…	…	-0.01
	Z_5	0.03	0.02	0.05	0.02	…	…	-0.01
	Z_6	0.02	0.03	0.02	-0.03	…	…	0.04
	Z_7	0.02	-0.03	0.01	0.01	…	…	0.04
	Z_8	0.01	-0.01	0.04	0.04	…	…	0.05
	Z_9	0.01	0.01	0.04	0.01	…	…	0.01
	Z_{10}	0.02	0.04	0.03	-0.01	…	…	0.03
	Z_{11}	-0.05	-0.01	0.04	-0.01	…	…	0.02
	Z_{12}	0.01	-0.03	0.02	0.02	…	…	0.01
	Z_{13}	0.01	0.01	-0.05	0.04	…	…	0.04
	Z_{14}	0.01	0.01	0.04	0.02	…	…	0.04
	Z_{15}	0.04	0.01	-0.01	0.03	…	…	0.04

Selanjutnya, menghitung nilai v_j dengan Persamaan (2-6). Sehingga, diperoleh nilai v_j sebagai berikut :

$$\|v_1\| = \sqrt{(-0.04)^2 + (-0.01)^2 + 0.01^2 + 0.02^2 + ... + 0.02^2} = 0.1296$$

$$\|v_2\| = \sqrt{0.02^2 + (-0.03)^2 + 0.01^2 + (-0.01)^2 + ... + (-0.02)^2} = 0.1288$$

$$\|v_3\| = \sqrt{(-0.03)^2 + (-0.03)^2 + 0.04^2 + 0.04^2 + ... + 0.01^2} = 0.1345$$

… … …

… … …

… … …

$$\|v_{15}\| = \sqrt{0.04^2 + 0.01^2 + (-0.01)^2 + 0.03^2 + ... + 0.04^2} = 0.133$$

Nilai hasil inisialisasi v_j ditunjukkan pada Tabel 3.10.

Tabel 3. 10 Tabel hasil inisialisasi nilai v_j

j	v_j
1	0.1296
2	0.1288
3	0.1345
4	0.1323
5	0.1371
6	0.1404
7	0.1311
8	0.1311
9	0.1371
10	0.1442
11	0.1330
12	0.1082
13	0.1292
14	0.1281
15	0.1330

Selanjutnya, menginisialisasi kembali bobot v_{ij} dengan menggunakan bobot v_{ij}*(lama)* pada Tabel 3.9 menggunakan Persamaan (2-7). Sehingga, diperoleh nilai bobot v_{ij} sebagai berikut :

$$v_{11} = \frac{0.7963 \times (-0.04)}{\|0.1296\|} = -0.2458$$

$$v_{12} = \frac{0.7963 \times (0.02)}{\|0.1288\|} = 0.1236$$

...
...
...

$$v_{21.15} = \frac{0.7963 \times (0.04)}{\|0.1330\|} = 0.2394$$

Nilai hasil inisialisasi kembali v_{ij} menggunakan bobot v_{ij}*(lama)* ditunjukkan pada Tabel 3.11.

Tabel 3. 11 Tabel nilai bobot baru v_{ij}

v_{ij}		I						
		x_1	x_2	x_3	x_4	$x_{...}$	$x_{...}$	x_{21}
j	z_1	-0.2458	0.1229	0.0614	0.1229	…	…	0.1229
	z_2	0.1236	-0.1854	0.0618	-0.0618	…	…	-0.1236
	z_3	-0.1776	-0.1776	0.2368	0.1776	…	…	0.0592
	z_4	0.2408	-0.1204	0.0602	0.0602	…	…	-0.0602
	z_5	0.1742	0.1162	0.2904	0.1162	…	…	-0.0581
	z_6	0.1135	0.1702	0.1135	-0.1702	…	…	0.2269
	z_7	0.1214	-0.1822	0.0607	0.0607	…	…	0.2429
	z_8	0.0607	-0.0607	0.2429	0.2429	…	…	0.3036
	z_9	0.0581	0.0581	0.2323	0.0581	…	…	0.0581
	z_{10}	0.1104	0.2209	0.1657	-0.0552	…	…	0.1657
	z_{11}	-0.2993	-0.0599	0.2394	-0.0599	…	…	0.1197
	z_{12}	0.0736	-0.2209	0.1472	0.1472	…	…	0.0736
	z_{13}	0.0616	0.0616	-0.3081	0.2465	…	…	0.2465
	z_{14}	0.0622	0.0622	0.2487	0.1244	…	…	0.2487
	z_{15}	0.2394	0.0599	-0.0599	0.1796	…	…	0.2394

Selanjutnya, menentukan nilai bias yang dipakai sebagai inisialisasi v_{0j}, yaitu bilangan acak antara β dan – β, sehingga diperoleh nilai bias yang ditunjukkan pada Tabel 3.12.

Tabel 3. 12 Tabel nilai bias awal v_{0j}

v_{0j}		0
		1
j	z_1	-0.7963
	z_2	-0.6567
	z_3	-0.7456
	z_4	-0.9546
	z_5	0.2786
	z_6	0.3637
	z_7	0.3568
	z_8	0.4563
	z_9	0.4557
	z_{10}	0.6568
	z_{11}	0.6668
	z_{12}	0.6778
	z_{13}	0.7235
	z_{14}	0.7357
	z_{15}	0.7567

Selanjutnya, menentukan nilai bobot dan bias awal yang dipakai sebagai inisialisasi bobot antara *hidden layer* ke *output layer* (w_{jk}) dan bias ke *output layer* (w_{0k}), yaitu dengan bilangan acak antara -0.05 hingga 0.05, sehingga diperoleh nilai bobot dan bias awal yang ditunjukkan pada Tabel 3.13 dan Tabel 3.14.

Tabel 3. 13 Tabel nilai bobot awal w_{jk}

w_{jk}		k
		y
j	Z_1	-0.04
	Z_2	0.03
	Z_3	-0.01
	Z_4	0.02
	Z_5	0.03
	Z_6	0.04
	Z_7	0.03
	Z_8	0.05
	Z_9	-0.01
	Z_{10}	-0.03
	Z_{11}	0.05
	Z_{12}	-0.02
	Z_{13}	0.04
	Z_{14}	-0.05
	Z_{15}	0.02

Tabel 3. 14 Tabel nilai bias awal w_{0k}

w_{0k}		k
		y
0	1	-0.04

Proses *Feedforward*

Setelah dilakukan langkah-langkah inisialisasi dengan menggunakan algoritma Nguyen-Widrow seperti di atas, maka selanjutnya dilakukan perhitungan tahapan pelatihan JST dengan menggunakan algoritma *feedforward*. Pada proses feedforward ini akan diberikan contoh perhitungan manual pada data ke-1 saja.

1. Menghitung sinyal masukan tiap unit pada *hidden layer*

Langkah ini menggunakan Persamaan (2-13). Sehingga, diperoleh nilai sinyal masukan tiap unit pada *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.15:

$$z\_in_1 = -0.7963 + (0.9)(-0.2458) + (0.9)(0.1229) + (0.9)(0.0614) + ... + (0.7)(0.1229) = 0.0454$$

$$z\_in_2 = -0.6567 + (0.9)(0.1236) + (0.9)(-0.1854) + (0.9)(0.0618) + ... + (0.7)(-0.1236) = -0.4342$$

$$z\_in_3 = -0.7456 + (0.9)(-0.1776) + (0.9)(-0.1776) + (0.9)(0.2368) + ... + (0.7)(0.0592) = -0.7871$$

...

...

...

$$z\_in_{15} = 0.7567 + (0.9)(0.2394) + (0.9)(0.0599) + (0.9)(-0.0599) + ... + (0.7)(0.2394) = 2.6901$$

Tabel 3. 15 Tabel nilai sinyal masukan pada hidden layer (z_in_j)

j	z_in_j
1	0.0454
2	-0.4342
3	-0.7871
4	0.5083
5	2.1604
6	1.0162
7	0.8668
8	0.6385
9	1.7915
10	2.8875
11	1.1875
12	1.8337
13	1.6294
14	2.4644
15	2.6901

2. Menghitung sinyal keluaran tiap unit pada *hidden layer* dengan fungsi aktivasi

Fungsi aktivasi yang digunakan adalah fungsi aktivasi sigmoid biner. Langkah ini menggunakan Persamaan (2-14). Sehingga, diperoleh nilai sinyal keluaran tiap unit pada *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.16:

$$z_1 = \frac{1}{1+e^{-0.0454}} = 0.5114$$

$$z_2 = \frac{1}{1+e^{-(-0.4342)}} = 0.3931$$

$$z_3 = \frac{1}{1+e^{-(-0.7871)}} = 0.3128$$

...

...

...

$$z_{15} = \frac{1}{1+e^{-2.6901}} = 0.9364$$

Tabel 3. 16 Tabel nilai sinyal keluaran pada hidden layer (z_j)

j	z_j
1	0.5114
2	0.3931
3	0.3128
4	0.6244
5	0.8966
6	0.7342
7	0.7041
8	0.6544
9	0.8571
10	0.9472
11	0.7663
12	0.8622
13	0.8361
14	0.9216
15	0.9364

3. Menghitung sinyal masukan tiap unit pada *output layer*

   Langkah ini menggunakan Persamaan (2-15). Sehingga, diperoleh nilai sinyal masukan tiap unit pada *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.17:

$$y\_in_1 = -0.04 + (-0.04)(0.5114) + (0.03)(0.3931) + (-0.01)(0.3128) + ... + (0.02)(0.9364) = 0.0610$$

Tabel 3. 17 Tabel nilai sinyal masukan pada output layer (y_in_k)

k	y_in_k
1	0.0610

4. Menghitung sinyal keluaran tiap unit pada *output layer*

   Langkah ini menggunakan Persamaan (2-16). Sehingga, diperoleh nilai sinyal masukan tiap unit pada *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.18:

$$y_1 = \frac{1}{1+e^{-0.0610}} = 0.5152$$

Tabel 3. 18 Tabel nilai sinyal keluaran pada *output layer* (y_k)

k	y_k
1	0.5152

Proses Info Error *Resilient Backpropagation*

Setelah dilakukan perhitungan pada proses *feedforward*, maka selanjutnya dilakukan proses perhitungan info *error resilient backpropagation*. Pada proses perhitungan info *error resilient backpropagation* ini akan diberikan contoh perhitungan manual pada data ke-1 saja.

a. Menghitung nilai informasi *error* (δ_k) pada tiap unit di *output layer*

   Langkah ini menggunakan Persamaan (2-17). Sehingga, diperoleh nilai informasi *error* pada unit output sebagai berikut dan ditunjukkan pada Tabel 3.19:

$$\delta_1 = (0.5152 \times (1 - 0.5152)) \times \frac{1}{2} \times (0.7 - 0.5152) = 0.0231$$

Tabel 3. 19 Tabel nilai informasi *error* pada *output layer* (δ_k)

k	δ
1	0.0231

b. Menghitung nilai informasi *error* (δ_j) pada tiap unit di *hidden layer*

Langkah ini menggunakan Persamaan (2-18). Sehingga, diperoleh nilai informasi *error* pada unit *hidden* sebagai berikut dan ditunjukkan pada Tabel 3.20:

$\delta_1 = (0.5114 \times (1 - 0.5114)) \times ((0.0231)(-0.04)) = -0.00023061$

$\delta_2 = (0.3931 \times (1 - 0.3931)) \times ((0.0231)(0.03)) = 0.000165142$

$\delta_3 = (0.3128 \times (1 - 0.3128)) \times ((0.0231)(-0.01)) = -0.000049597$

...

...

...

$\delta_{15} = (0.9364 \times (1 - 0.9364) \times ((0.0231 \times 0.03)) = 0.0000274661$

Tabel 3. 20 Tabel nilai informasi *error* pada *hidden layer* (δ_j)

j	δ
1	-0.00023061
2	0.000165142
3	-0.000049597
4	0.000108225
5	0.0000641518
6	0.000180096
7	0.000144218
8	0.000260909
9	-0.000028258
10	-0.000034602
11	0.000206605
12	-0.000054828
13	0.000126487
14	-0.00008335
15	0.0000274661

Proses Hitung *Gradient* dan Perbaikan Bobot

Setelah dilakukan perhitungan pada proses info *error resilient backpropagation*, maka selanjutnya dilakukan perhitungan pada proses hitung *gradient*. Pada proses hitung *gradient* ini akan diberikan contoh perhitungan manual pada data ke-1 saja.

a. Menghitung nilai turunan pertama fungsi *error* terhadap bobot pada *hidden* ke *output layer* $\left(\frac{\partial E}{\partial w_{jk}}^{(t)}\right)$

Langkah ini menggunakan Persamaan (2-20). Sehingga, diperoleh turunan pertama fungsi error terhadap bobot *hidden* ke *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.21 sampai Tabel 3.22:

$$\frac{\partial E}{\partial w_{11}} = 0 + ((0.0231)(0.5114)) = 0.0118$$

$$\frac{\partial E}{\partial w_{21}} = 0 + ((0.0231)(0.3931)) = 0.0091$$

$$\frac{\partial E}{\partial w_{31}} = 0 + ((0.0231)(0.3128)) = 0.0072$$

...

...

...

$$\frac{\partial E}{\partial w_{15.1}} = 0 + ((0.0231)(0.9364)) = 0.0216$$

Tabel 3. 21 Tabel nilai turunan pertama fungsi *error* pada w_{jk}

$\left(\frac{\partial E}{\partial w_{jk}}^{(t)}\right)$		**k**
		y
j	$\mathbf{Z_1}$	0.0118
	$\mathbf{Z_2}$	0.0091
	$\mathbf{Z_3}$	0.0072
	$\mathbf{Z_4}$	0.0144
	$\mathbf{Z_5}$	0.0207
	$\mathbf{Z_6}$	0.0169
	$\mathbf{Z_{...}}$	…
	$\mathbf{Z_{...}}$	…
	$\mathbf{Z_{15}}$	0.0216

Tabel 3. 22 Tabel nilai turunan pertama fungsi error pada w_{0k}

$\left(\frac{\partial E}{\partial w_{0k}}^{(t)}\right)$		**k**
		y
j	**1**	0.0231

b. Menghitung nilai turunan pertama fungsi *error* terhadap bobot pada input ke *hidden layer* $\left(\frac{\partial E}{\partial v_{ij}}^{(t)}\right)$

Langkah ini menggunakan Persamaan (2-19). Sehingga, diperoleh nilai turunan pertama fungsi *error* terhadap bobot pada input ke *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.23 sampai Tabel 3.24:

$$\frac{\partial E}{\partial v_{11}} = 0 + ((-0.00023061)(0.9)) = -0.000208$$

$$\frac{\partial E}{\partial v_{12}} = 0 + ((0.000165142)(0.9)) = 0.0001486$$

$$\frac{\partial E}{\partial v_{13}} = 0 + ((-0.000049597)(0.9)) = -4.46E-05 = -0.000046$$

...

...

...

$$\frac{\partial E}{\partial v_{21.15}} = 0 + ((0.0000274661)(0.7)) = 1.92263E-05 = 0.0000192263$$

Tabel 3.23 Tabel nilai turunan pertama fungsi *error* pada v_{ij}

$\left(\frac{\partial E}{\partial v_{ij}}^{(t)}\right)$		**i**						
		x_1	**x_2**	**x_3**	**x_4**	**$x_{...}$**	**$x_{...}$**	**x_{21}**
j	**Z_1**	-0.000208	-0.000208	-0.00021	-0.00021	…	…	-0.00016143
	Z_2	0.0001486	0.0001486	0.000149	0.000149	…	…	0.0001156
	Z_3	-4.46E-05	-4.46E-05	-4.5E-05	-4.5E-05	…	…	-3.47182E-05
	Z_4	9.74E-05	9.74E-05	9.74E-05	9.74E-05	…	…	7.57578E-05
	Z_5	5.774E-05	5.774E-05	5.77E-05	5.77E-05	…	…	4.49063E-05
	Z_6	0.0001621	0.0001621	0.000162	0.000162	…	…	0.000126068
	$Z_{...}$	…	…	…	…	…	…	…
	$Z_{...}$	…	…	…	…	…	…	…
	Z_{15}	2.47195E-05	2.47195E-05	2.47E-05	2.47E-05	…	…	1.92263E-05

Tabel 3.24 Tabel nilai turunan pertama fungsi *error* pada v_{0j}

$\left(\frac{\partial E}{\partial v_{0j}}\right)^{(t)}$		**0**
		1
j	**Z_1**	-0.00023061
	Z_2	0.00016514
	Z_3	-4.9597E-05
	Z_4	0.00010823
	Z_5	6.4152E-05
	Z_6	0.0001801
	$Z_{...}$	…
	$Z_{...}$	…
	Z_{15}	2.7466E-05

c. Menghitung nilai *learning rate* pada bobot antara *hidden* ke *output layer*

Langkah ini menggunakan Persamaan (2-21). Karena nilai $\frac{\partial E}{\partial w_{jk}}^{(t-1)} = 0$, maka nilai $\frac{\partial E}{\partial w_{jk}}^{(t-1)} * \frac{\partial E}{\partial w_{jk}}^{(t)} = 0$, sedangkan untuk nilai $\frac{\partial E}{\partial w_{jk}}^{(t)}$ merupakan nilai *gradient* sampai data terakhir, yaitu sampai data ke-4. Nilai $\frac{\partial E}{\partial w_{jk}}^{(t)}$ dan $\frac{\partial E}{\partial w_{0k}}^{(t)}$ sampai data ke-4 ditunjukkan pada Tabel 3.25 dan Tabel 3.26.

Tabel 3.25 Tabel nilai *gradient* saat ini terhadap bobot w_{jk}

$\frac{\partial E}{\partial w_{jk}}^{(t)}$		**k**
		y
j	**Z_1**	0.00848
	Z_2	0.00694
	Z_3	0.00631
	Z_4	0.01085
	Z_5	0.01573
	Z_6	0.01269
	$Z_{...}$	…
	$Z_{...}$	…
	Z_{15}	0.01619

Tabel 3.26 Tabel nilai *gradient* saat ini terhadap bobot w_{0k}

$\frac{\partial E}{\partial w_{0k}}^{(t)}$		**k**
		y
0	**1**	0.01741

Selanjutnya diperoleh nilai *learning rate* pada bobot antara *hidden* ke *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.27 dan Tabel 3.28:

$\Delta_{11} = 0.1$
$\Delta_{21} = 0.1$
$\Delta_{31} = 0.1$
...
...
...
$\Delta_{15.1} = 0.1$

Tabel 3.27 Tabel nilai *learning rate* (Δ_{jk}) pada w_{jk}

Δ_{jk}		**k**
		y
j	Z_1	0.1
	Z_2	0.1
	Z_3	0.1
	Z_4	0.1
	Z_5	0.1
	Z_6	0.1
	$Z_{...}$	...
	$Z_{...}$	...
	Z_{15}	0.1

Tabel 3.28 Tabel nilai *learning rate* (Δ_{0k}) pada w_{0k}

Δ_{0k}		**k**
		y
0	**1**	0.1

d. Perbarui nilai bobot antara *hidden* ke *output layer*

Perhitungan perubahan nilai bobot antara *hidden* ke *output layer* dilakukan setelah semua pola dihitung (dilakukan pada setiap iterasi). Karena nilai

$\frac{\partial E}{\partial w_{jk}}^{(t)}$ dan $\frac{\partial E}{\partial w_{0k}}^{(t)}$ bernilai positif semua (Tabel 3.25 dan Tabel 3.26), maka nilai perubahan bobot ditingkatkan dengan memberi tanda positif. Langkah ini menggunakan Persamaan (2-22), sehingga diperoleh nilai perubahan bobot pada bobot antara *hidden* ke *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.29 dan Tabel 3.30:

$\Delta w_{11} = 0.1$
$\Delta w_{21} = 0.1$
$\Delta w_{31} = 0.1$
...
...
...
$\Delta w_{15.1} = 0.1$

Tabel 3.29 Tabel nilai perubahan bobot (Δw_{jk})

Δw_{jk}		k
		y
j	Z_1	0.10
	Z_2	0.10
	Z_3	0.10
	Z_4	0.10
	Z_5	0.10
	Z_6	0.10
	$Z_{...}$	...
	$Z_{...}$	...
	Z_{15}	0.10

Tabel 3.30 Tabel nilai perubahan bobot (Δw_{0k})

Δw_{0k}		k
		y
0	1	0.10

Setelah semua pola pada data latih dihitung, maka diperoleh nilai bobot baru antara *hidden* ke *output layer* dengan menggunakan Persamaan (2-23) sebagai berikut dan ditunjukkan pada Tabel 3.31 dan Tabel 3.32:

$w_{11}baru = -0.04 + 0.10 = 0.06$
$w_{21}baru = 0.03 + 0.10 = 0.13$
$w_{31}baru = -0.01 + 0.10 = 0.09$
...
...
...
$w_{15.1}baru = 0.02 + 0.10 = 0.12$

Tabel 3. 31 Tabel nilai bobot baru *hidden* ke *output layer* (w_{jk})

w_{jk}(baru)		**k**
		y
j	**Z_1**	0.06
	Z_2	0.13
	Z_3	0.09
	Z_4	0.12
	Z_5	0.13
	Z_6	0.14
	$Z_{...}$	...
	$Z_{...}$	...
	Z_{15}	0.12

Tabel 3. 32 Tabel nilai bobot baru bias ke *output layer* (w_{0k})

w_{0k}(baru)		**k**
		y
0	**1**	0.06

e. Menghitung nilai *learning rate* pada bobot antara input ke *hidden layer*

Langkah ini menggunakan Persamaan (2-21). Karena nilai $\frac{\partial E}{\partial v_{ij}}^{(t-1)} = 0$, maka nilai $\frac{\partial E}{\partial v_{ij}}^{(t-1)} * \frac{\partial E}{\partial v_{ij}}^{(t)} = 0$, sedangkan untuk nilai $\frac{\partial E}{\partial v_{ij}}^{(t)}$ merupakan nilai *gradient* sampai data terakhir, yaitu sampai data ke-4. Nilai $\frac{\partial E}{\partial v_{ij}}^{(t)}$ dan $\frac{\partial E}{\partial v_{0j}}^{(t)}$ sampai data ke-4 ditunjukkan pada Tabel 3.33 dan Tabel 3.34.

Tabel 3.33 Tabel nilai *gradient* saat ini terhadap bobot v_{ij}

$\frac{\partial E}{\partial v_{ij}}^{(t)}$		i					
		x_1	x_2	x_3	$x_{...}$	$x_{...}$	x_{21}
j	Z_1	-0.000156447	-0.000156447	-0.00026	…	…	-0.00016024
	Z_2	0.000112747	0.000112747	0.000189	…	…	0.000115456
	Z_3	-3.66528E-05	-3.66528E-05	-5.9E-05	…	…	-3.749E-05
	Z_4	7.36284E-05	7.36284E-05	0.000124	…	…	7.54236E-05
	Z_5	4.11964E-05	4.11964E-05	7.39E-05	…	…	4.23311E-05
	Z_6	0.000123809	0.000123809	0.000209	…	…	0.000126879
	$Z_{...}$	…	…	…	…	…	…
	$Z_{...}$	…	…	…	…	…	…
	Z_{15}	2.04647E-05	2.04647E-05	3.3E-05	…	…	2.09405E-05

Tabel 3.34 Tabel nilai *gradient* saat ini terhadap bobot v_{0j}

v_{0j}		0
		1
j	Z_1	-0.00017383
	Z_2	0.00012527
	Z_3	-4.0725E-05
	Z_4	8.1809E-05
	Z_5	4.5774E-05
	Z_6	0.00013757
	$Z_{...}$	…
	$Z_{...}$	…
	Z_{15}	2.2739E-05

Selanjutnya diperoleh nilai *learning rate* pada bobot antara input ke *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.35 dan Tabel 3.36:

$\Delta_{11} = 0.1$

$\Delta_{21} = 0.1$

$\Delta_{31} = 0.1$

...

...

...

$\Delta_{15.12} = 0.1$

Tabel 3.35 Tabel nilai *learning rate* (Δ_{ij}) pada v_{ij}

Δ_{ij}		I						
		x_1	x_2	x_3	x_4	$x_{...}$	$x_{...}$	x_{21}
j	Z_1	0.1	0.1	0.1	0.1	…	…	0.1
	Z_2	0.1	0.1	0.1	0.1	…	…	0.1
	Z_3	0.1	0.1	0.1	0.1	…	…	0.1
	Z_4	0.1	0.1	0.1	0.1	…	…	0.1
	Z_5	0.1	0.1	0.1	0.1	…	…	0.1
	Z_6	0.1	0.1	0.1	0.1	…	…	0.1
	$Z_{...}$	…	…	…	…	…	…	…
	$Z_{...}$	…	…	…	…	…	…	…
	Z_{15}	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Tabel 3.36 Tabel nilai *learning rate* (Δ_{0j}) pada v_{0j}

Δ_{0j}		i
		1
j	Z_1	0.1
	Z_2	0.1
	Z_3	0.1
	Z_4	0.1
	Z_5	0.1
	Z_6	0.1
	$Z_{...}$	…
	$Z_{...}$	…
	Z_{15}	0.1

f. Perbarui nilai bobot antara *input* ke *hidden layer*

Perhitungan perubahan nilai bobot antara *input* ke *hidden layer* dilakukan setelah semua pola dihitung (dilakukan pada setiap iterasi). Langkah ini menggunakan Persamaan (2-22) berdasarkan pada nilai *gradient* saat yang ditunjukkan pada Tabel 3.33 dan Tabel 3.34. Sehingga, diperoleh nilai perubahan bobot pada bobot antara input ke *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.37 dan Tabel 3.38:

$\Delta v_{11} = -0.10$

$\Delta v_{21} = -0.10$

$\Delta v_{31} = -0.10$

...

...

...

$\Delta v_{21.15} = 0.10$

Tabel 3.37 Tabel nilai perubahan bobot (Δv_{ij})

Δv_{ij}		I						
		x_1	x_2	x_3	x_4	$x_{...}$	$x_{...}$	x_{21}
j	Z_1	-0.10	-0.10	-0.10	-0.10	…	…	-0.10
	Z_2	0.10	0.10	0.10	0.10	…	…	0.10
	Z_3	-0.10	-0.10	-0.10	-0.10	…	…	-0.10
	Z_4	0.10	0.10	0.10	0.10	…	…	0.10
	Z_5	0.10	0.10	0.10	0.10	…	…	0.10
	Z_6	0.10	0.10	0.10	0.10	…	…	0.10
	$Z_{...}$	…	…	…	…	…	…	…
	$Z_{...}$	…	…	…	…	…	…	…
	Z_{15}	0.10	0.10	0.10	0.10	…	…	0.10

Tabel 3.38 Tabel nilai perubahan bobot (Δv_{0j})

Δv_{0j}		0
		1
j	Z_1	-0.10
	Z_2	0.10
	Z_3	-0.10
	Z_4	0.10
	Z_5	0.10
	Z_6	0.10
	$Z_{...}$	…
	$Z_{...}$	…
	Z_{15}	0.10

Setelah semua pola pada data latih dihitung, maka diperoleh nilai bobot baru antara *input* ke *hidden layer* dengan menggunakan Persamaan (2-23) sebagai berikut dan ditunjukkan pada Tabel 3.39 dan Tabel 3.40:

$v_{11}baru = -0.2458 + (-0.10) = -0.3458$

$v_{21}baru = 0.1236 + (-0.10) = 0.0229$

$v_{31}baru = -0.1776 + (-0.10) = -0.0386$

.

.

.

$v_{21.15}baru = 0.2394 + 0.10 = 0.3394$

Tabel 3. 39 Tabel nilai bobot baru *input* ke *hidden layer* (v_{ij})

v_{ij}(baru)		I						
		x_1	x_2	x_3	x_4	$x_{...}$	$x_{...}$	x_{21}
j	Z_1	-0.3458	0.0229	-0.0386	0.0229	…	…	0.0229
	Z_2	0.2236	-0.0854	0.1618	0.0382	…	…	-0.0236
	Z_3	-0.2776	-0.2776	0.1368	0.0776	…	…	-0.0408
	Z_4	0.3408	-0.0204	0.1602	0.1602	…	…	0.0398
	Z_5	0.2742	0.2162	0.3904	0.2162	…	…	0.0419
	Z_6	0.2135	0.2702	0.2135	-0.0702	…	…	0.3269
	$Z_{...}$	…	…	…	…	…	…	…
	$Z_{...}$	…	…	…	…	…	…	…
	Z_{15}	0.3394	0.1599	0.0401	0.2796	…	…	0.3394

Tabel 3. 40 Tabel nilai bobot baru bias ke *hidden layer* (v_{0j})

v_{0j}(baru)		0
		1
j	Z_1	-0.8963
	Z_2	-0.5567
	Z_3	-0.8456
	Z_4	-0.8546
	Z_5	0.3786
	Z_6	0.4637
	$Z_{...}$	
	$Z_{...}$	
	Z_{15}	0.8567

Hasil dari pelatihan JST ini berupa nilai bobot yang akan dipakai dalam proses pengujian JST, sehingga nilai bobot yang akan dipakai pada proses pengujian dapat ditunjukkan pada Tabel 3.31, Tabel 3.32, Tabel 3.39, dan Tabel 3.40.

Proses Pengujian JST dengan Algoritma *Feedforward*

Pada pengujian JST *Resilient Backpropagation* ini yang akan dijadikan pengujian adalah data ke-1 saja dengan menggunakan algoritma *feedforward* seperti pada proses pelatihan. Hasil klasifikasi akan dianggap benar, apabila nilai hasil klasifikasi sesuai dengan nilai yang ada pada target. Langkah-langkah pengujian JST dengan algoritma *feddforward* sebagai berikut :

a. Menginisialisasi bobot v_{ij} dan bobot w_{jk} hasil dari proses pelatihan

Diperoleh nilai bobot antara *input* dengan *hidden layer* (v_{ij}) dan bobot antara *hidden* dengan *output layer* (w_{jk}) yang ditunjukkan pada Tabel 3.31, Tabel 3.32, Tabel 3.39, dan Tabel 3.40.

b. Menghitung sinyal masukan tiap unit pada *hidden layer*

Langkah ini menggunakan Persamaan (2-13). Sehingga, diperoleh nilai sinyal masukan tiap unit pada *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.41:

$$z\_in_1 = -0.8963 + (0.9)(-0.3458) + (0.9)(0.0229) + (0.9)(-0.0386) + ... + (0.7)(0.0229) = -1.9552$$

$$z\_in_2 = -0.5567 + (0.9)(0.2236) + (0.9)(-0.0854) + (0.9)(0.1618) + ... + (0.7)(-0.0236) = 1.4434$$

$$z\_in_3 = -0.8456 + (0.9)(-0.2776) + (0.9)(-0.2776) + (0.9)(0.1368) + ... + (0.7)(-0.0408) = -2.6060$$

...

...

...

$$z\_in_{15} = 0.8567 + (0.9)(0.3394) + (0.9)(0.1599) + (0.9)(0.0401) + ... + (0.7)(0.3394) = 4.5203$$

Tabel 3. 41 Tabel nilai sinyal masukan pada hidden layer (z_in_j)

j	z_in_j
1	-1.9552
2	1.4434
3	-2.6060
4	2.3621
5	3.9575
6	2.8268
7	2.7811
8	2.5892
9	-0.0288
10	0.9644
11	3.0296
12	-0.0122
13	3.4208
14	0.4749
15	4.5203

c. Menghitung sinyal keluaran tiap unit pada *hidden layer* dengan fungsi aktivasi Fungsi aktivasi yang digunakan adalah fungsi aktivasi sigmoid biner. Langkah ini menggunakan Persamaan (2-14). Sehingga, diperoleh nilai sinyal keluaran tiap unit pada *hidden layer* sebagai berikut dan ditunjukkan pada Tabel 3.42:

$$z_1 = \frac{1}{1+e^{-(-1.9552)}} = 0.1240$$

$$z_2 = \frac{1}{1+e^{-1.4434}} = 0.8090$$

$$z_3 = \frac{1}{1+e^{-(-2.6060)}} = 0.0688$$

...

...

...

$$z_{15} = \frac{1}{1+e^{-4.5203}} = 0.9892$$

Tabel 3. 42 Tabel nilai sinyal keluaran pada *hidden layer* (z_j)

j	z_j
1	0.1240
2	0.8090
3	0.0688
4	0.9139
5	0.9812
6	0.9441
7	0.9416
8	0.9302
9	0.4928
10	0.7240
11	0.9539
12	0.4970
13	0.9683
14	0.6165
15	0.9892

1. Menghitung sinyal masukan tiap unit pada *output layer*

   Langkah ini menggunakan Persamaan (2-15). Sehingga, diperoleh nilai sinyal masukan tiap unit pada *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.43 :

   $$y\_in_1 = 0.06 + (0.06)(0.1240) + (0.13)(0.8090) + (0.09)(0.0688) + ... + (0.12)(0.9892)$$

   $$= 1.3731$$

Tabel 3. 43 Tabel nilai sinyal masukan pada *output layer* (y_in_k)

k	y_in_k
1	1.3731

2. Menghitung sinyal keluaran tiap unit pada *output layer*

   Langkah ini menggunakan Persamaan (2-16). Sehingga, diperoleh nilai sinyal masukan tiap unit pada *output layer* sebagai berikut dan ditunjukkan pada Tabel 3.44 :

   $$y_1 = \frac{1}{1 + e^{-1.3731}} = 0.7979$$

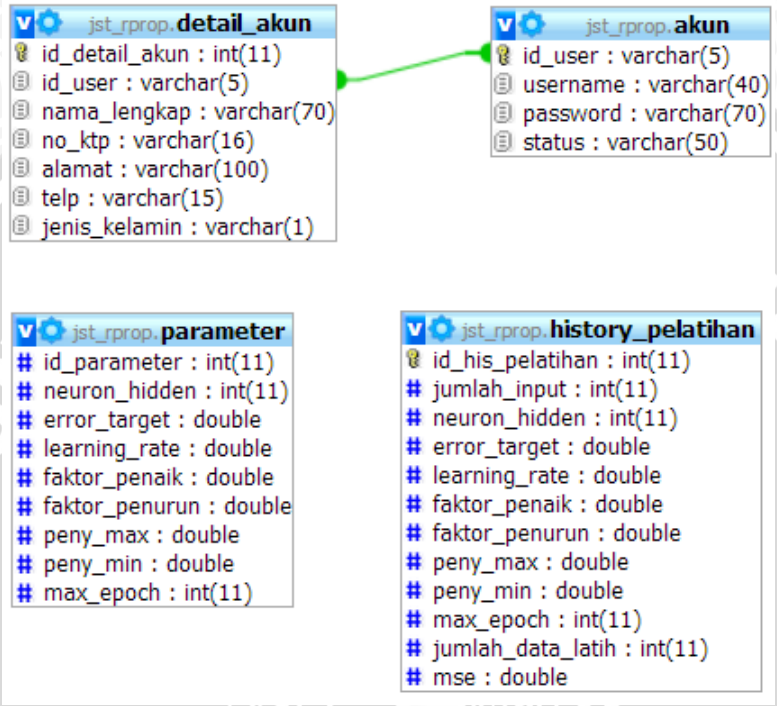
Tabel 3. 44 Tabel nilai sinyal keluaran pada *output layer* (y_k)

k	y_k
1	0.7979

Nilai keluaran yang dihasilkan = 0.7979, nilai ini berada pada rentang target $0.6 \leq y_k < 0.8$, maka output hasil pengujian pada data pola ke-1 masuk ke dalam kategori Keluarga Sejahtera III. Nilai keluaran yang dihasilkan ini bisa disesuaikan dengan tabel konversi nilai input target pada Tabel 3.2. Hal ini menunjukkan bahwa hasil prediksi data pola ke-1 dengan menggunakan JST *resilient backpropagation* termasuk ke dalam Keluarga Sejahtera III (KS III), sedangkan nilai target pada data asli sebesar 0.7 yang tergolong ke dalam KS III. Hasil prediksi data pola ke-1 sama dengan target pada data asli, sehingga hasil prediksi bisa dikatakan "Benar", karena nilai keluaran masih berada pada rentang nilai yang dikehendaki.

3.2.9 Perancangan Basis Data

Sistem yang akan dibuat memerlukan database untuk menyimpan seluruh data yang digunakan oleh sistem. Tabel yang dapat disusun berdasarkan kebutuhan sistem in, yaitu tabel akun, tabel detail_akun, tabel parameter, dan tabel history_pelatihan. Masing-masing tabel tersebut ditunjukkan pada Gambar 3.15.



Gambar 3. 15 Perancangan basis data

Struktur dari masing-masing tabel dapat dijelaskan secara terperinci pada Tabel 3.45 sampai Tabel 3.48 sebagai berikut :

Tabel 3. 45 Tabel Akun

No.	Nama Atribut	Tipe Data	Keterangan
1	id_user	Varchar (5)	*Primary key, Notnull*
2	username	Varchar (40)	*Notnull*
3	password	Varchar (70)	*Notnull*
4	Status	Varchar (50)	*Notnull*

Keterangan :

- Fungsi tabel : menyimpan data akun administrator.
- id_user : informasi tentang nomor identitas admin.

- username : informasi tentang username yang digunakan oleh admin.
- password : informasi tentang password yang digunakan oleh admin.
- Status : informasi tentang status administrator.

Tabel 3. 46 Tabel Detail_akun

No.	Nama Atribut	Tipe Data	Keterangan
1	id_detail_akun	Int	*Primary key, Notnull*
2	id_user	Varchar (5)	*Foreign key*
3	nama_lengkap	Varchar (70)	*Notnull*
4	no_ktp	Varchar (16)	*Notnull*
5	Alamat	Varchar (100)	*Notnull*
6	Telp	Varchar (15)	*Notnull*
7	jenis_kelamin	Varchar (1)	*Notnull*

Keterangan :

- Fungsi tabel : menyimpan data rincian akun administrator.
- id_detail_akun : informasi tentang nomor identitas detail akun admin.
- id_user : informasi tentang nomor identitas admin.
- nama_lengkap : informasi tentang nama lengkap admin.
- no_ktp : informasi tentang nomor ktp admin.
- alamat : informasi tentang alamat rumah admin.
- telp : informasi tentang nomor telepon admin.
- jenis_kelamin : informasi tentang jenis kelamin admin.

Tabel 3. 47 Tabel Parameter

No.	Nama Atribut	Tipe Data	Keterangan
1	id_parameter	Int	*Primary key, Notnull*
2	neuron_hidden	Int	*Notnull*
3	error_target	Double	*Notnull*
4	learning_rate	Double	*Notnull*
5	faktor_penaik	Double	*Notnull*
6	faktor_penurun	Double	*Notnull*
7	peny_max	Double	*Notnull*
8	peny_min	Double	*Notnull*
9	max_epoch	Int	*Notnull*

Keterangan :

- Fungsi tabel : menyimpan data parameter JST yang digunakan untuk tahap pelatihan JST *resilient backpropagation.*
- id_parameter : informasi tentang nomor identitas parameter JST.
- neuron_hidden : informasi tentang jumlah *neuron hidden*.
- error_target : informasi tentang *error target* untuk tahap pelatihan.
- learning_rate : informasi tentang *learning rate* untuk tahap pelatihan.
- faktor_penaik : informasi tentang faktor penaik untuk tahap pelatihan.
- faktor_penurun : informasi tentang faktor penurun untuk tahap pelatihan.
- peny_max : informasi tentang penyesuaian maksimum untuk tahap pelatihan.
- peny_min : informasi tentang penyesuaian minimum untuk tahap pelatihan.
- max_epoch : informasi tentang maksimum iterasi atau *epoch* yang digunakan untuk melatih JST.

Tabel 3. 48 Tabel History_pelatihan

No.	Nama Atribut	Tipe Data	Keterangan
1	id_pelatihan	Int	*Primary key, Notnull*
2	jumlah_input	Int	*Notnull*
3	neuron_hidden	Int	*Notnull*
4	errror_target	Double	*Notnull*
5	learning_rate	Double	*Notnull*
6	faktor_penaik	Double	*Notnull*
7	faktor_penurun	Double	*Notnull*
8	peny_max	Double	*Notnull*
9	peny_min	Double	*Notnull*
10	max_epoch	Int	*Notnull*
11	jumlah_data_latih	Int	*Notnull*
12	Mse	Double	*Notnull*

Keterangan :

- Fungsi tabel : menyimpan data riwayat dari pelatihan JST *resilient backpropagation* yang pernah dilakukan.
- id_pelatihan : informasi tentang nomor identitas pelatihan JST.

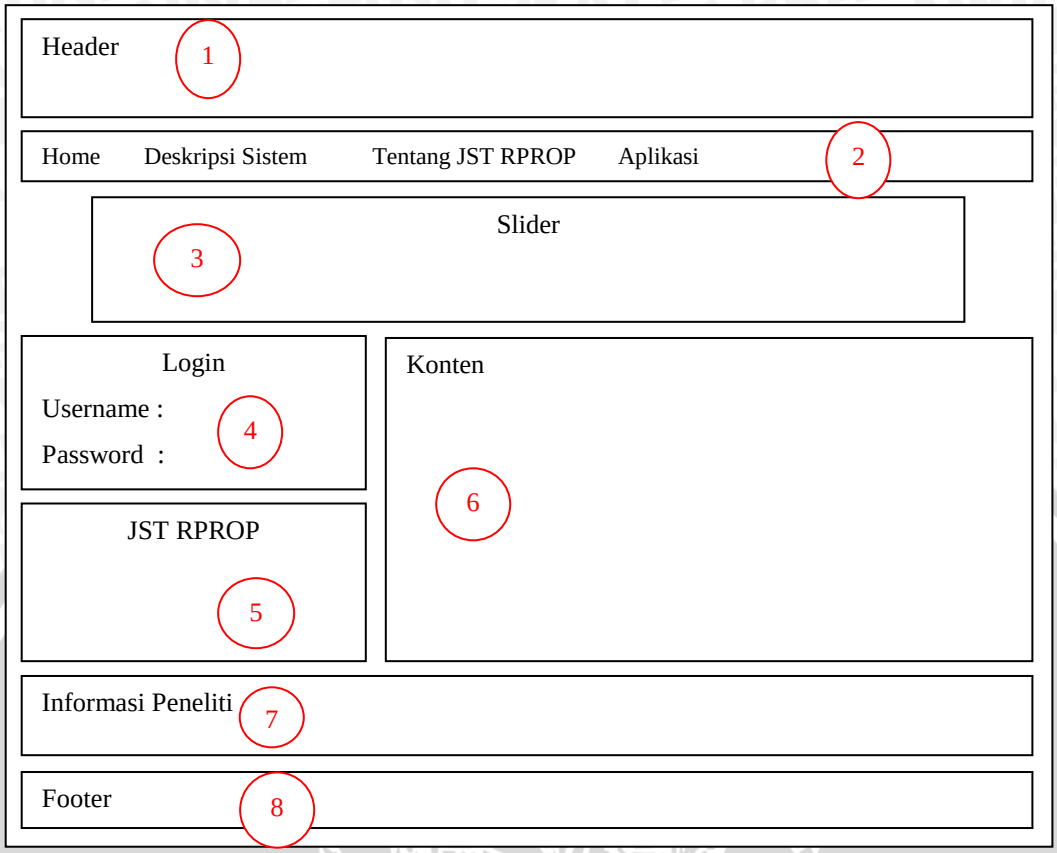
- jumlah_input : informasi tentang jumlah *neuron input* pada pelatihan.
- neuron_hidden : informasi tentang jumlah *neuron hidden* pada pelatihan.
- error_target : informasi tentang *error target* pada pelatihan.
- learning_rate : informasi tentang *learning rate* pada pelatihan.
- faktor_penaik : informasi tentang faktor penaik pada pelatihan.
- faktor_penurun : informasi tentang faktor penurun pada pelatihan.
- peny_max : informasi tentang penyesuaian maksimum pada pelatihan.
- peny_min : informasi tentang penyesuaian minimum pada pelatihan.
- max_epoch : informasi tentang maksimum iterasi atau *epoch* yang digunakan pada pelatihan.
- jumlah_data_latih : informasi tentang jumlah data latih yang digunakan pada saat pelatihan JST.
- mse : informasi tentang MSE optimal yang dapat dicapai pada saat pelatihan JST.

3.2.10 Perancangan Antarmuka

Perancangan antarmuka digunakan untuk merancang interaksi anatar pengguna dan sistem dengan tujuan mempermudah pengguna dalam aplikasi. Antarmuka pada sistem klasifikasi status tahapan keluarga sejahtera dengan menggunakan JST *resilient backpropagation* ini dibagi menjadi dua bagian, yaitu antarmuka utama dan antarmuka administrator. Berdasarkan hasil analisa secara keseluruhan terdapat beberapa bagian yang dibutuhkan dalam antarmuka sistem ini, yaitu:

1. Antarmuka utama

Antarmuka utama dapat dilihat pada Gambar 3.16 seperti di bawah ini.



Gambar 3. 16 Rancangan antarmuka utama

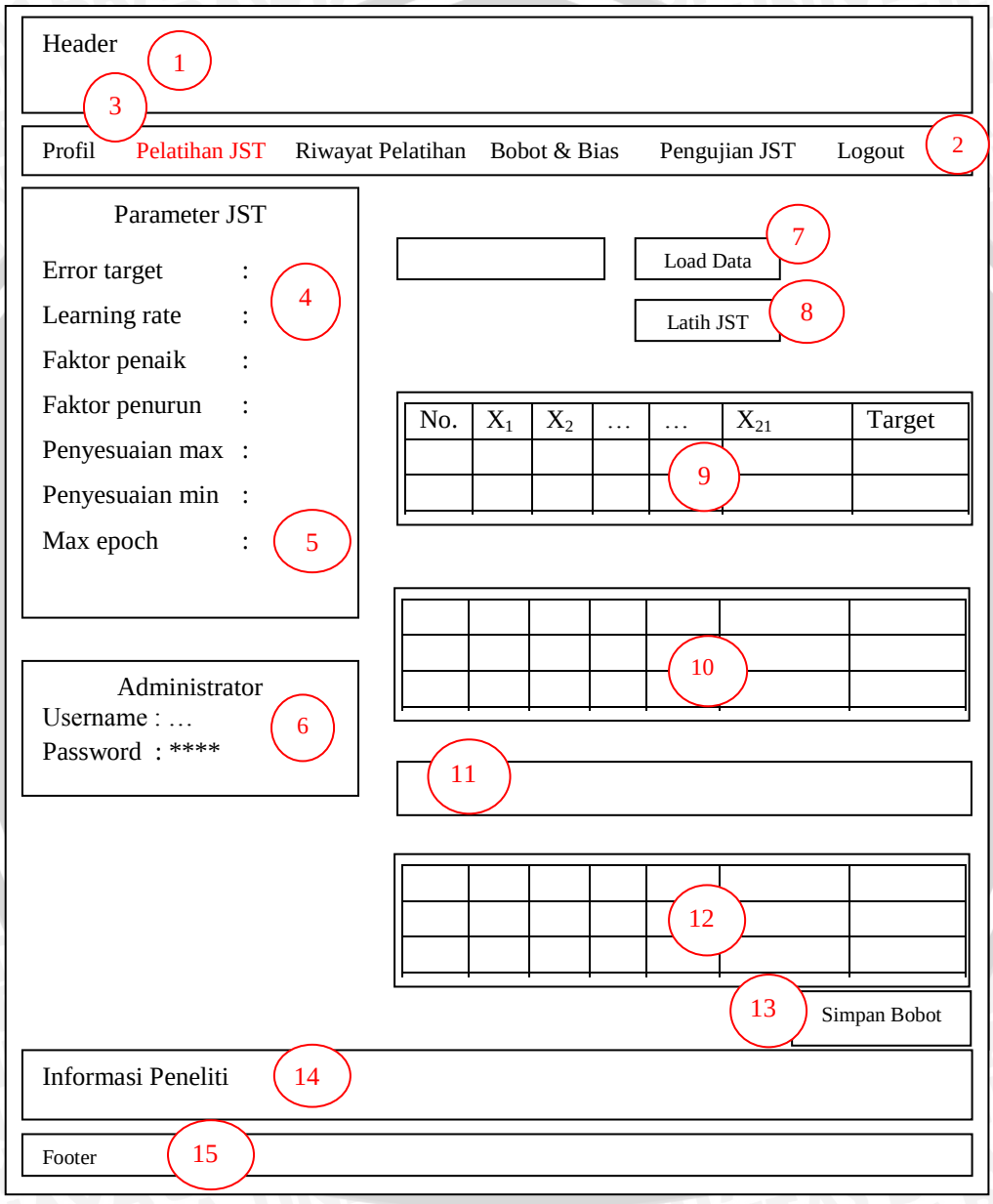
Keterangan :

1. *Header* sebagai informasi singkat mengenai sistem klasifikasi status tahapan keluarga sejahtera dengan JST *resilient backpropagation*.
2. Menu utama dari antarmuka.
3. *Slider* sebagai penambah nilai estetika sistem yang dibuat agar lebih informatif.
4. *Field* untuk *Login*, yaitu sarana untuk masuk ke dalam antarmuka administrator.
5. Sebagai tempat untuk menyajikan informasi sekilas mengenai metode yang digunakan dalam sistem, yaitu JST RPROP (*resilient backpropagation*).
6. Sebagai tempat untuk menyajikan konten berupa informasi yang ada pada menu *Home*, Deskripsi Sistem, dan Tentang JST RPROP.
7. Sebagai tempat untuk menyajikan informasi tentang peneliti.
8. *Footer* untuk menyajikan menu utama seperti halnya pada kotak no. 2.

2. Antarmuka administrator

Antarmuka administrator ini dapat dilihat pada Gambar 3.17 sampai 3.19 seperti yang tertera di bawah ini.

a. *Form* pelatihan JST



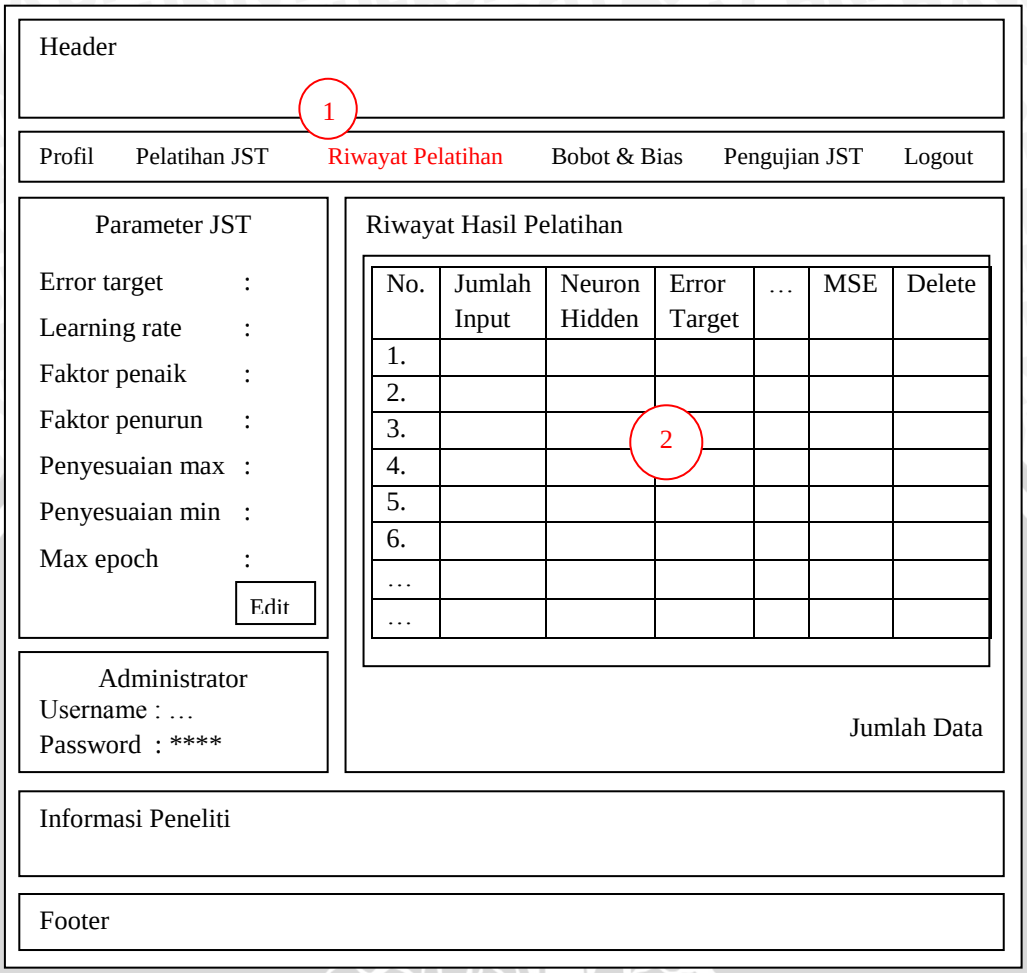
Gambar 3. 17 Rancangan *form* pelatihan JST

Keterangan :

1. Area *Header* sebagai area untuk menampilkan informasi sistem klasifikasi status tahapan keluarga sejahtera dengan JST *resilient backpropagation*.

2. Menu utama dari antarmuka administrator, yaitu :
   - "Profil" untuk menampilkan informasi mengenai data akun pribadi administrator.
   - "Pelatihan JST" untuk menampilkan *form* pelatihan JST.
   - "Riwayat Pelatihan" untuk menampilkan data riwayat pelatihan JST yang pernah dilakukan.
   - "Bobot dan Bias" untuk menampilkan data bobot dan bias terakhir yang disimpan dari hasil pelatihan JST.
   - "Pengujian JST" untuk menampilkan *form* pengujian JST.
   - "Logout" sebagai sarana keluar dari hak akses sebagai administrator.
3. Menandakan bahwa pengguna sedang berada pada *form* pelatihan JST setelah memilih menu "Pelatihan JST"
4. Area "Parameter JST" sebagai tempat untuk menampilkan nilai masing-masing parameter JST.
5. Tombol *"Edit"* untuk menampilan *form* edit untuk menyetting nilai dari parameter JST.
6. Area "Administrator" sebagai tempat untuk menginformasikan orang yang sedang mengakses masih berstatus login ke dalam sistem.
7. Tombol "*Load Data"* untuk memilih data yang akan dilatihkan dalam format csv.
8. Tombol "Latih JST" untuk memulai proses pelatihan JST berdasarkan data yang telah di-*load*.
9. Tabel untuk menyajikan data pelatihan yang telah di-*load* dan hasil dari pelatihan JST.
10. Tabel untuk menyajikan hasil inisialisasi bobot dan bias awal dengan algoritma Nguyen Widrow.
11. *Field* untuk menyajikan hasil MSE optimal pelatihan JST.
12. Tabel untuk menyajikan bobot dan bias akhir hasil dari pelatihan JST.
13. Tombol "Simpan Bobot" untuk menyimpan bobot dan bias akhir hasil pelatihan JST .
14. Sebagai tempat untuk menyajikan informasi kontak dari peneliti.
15. Area *Footer* untuk menyajikan menu utama seperti pada kotak no. 2.

b. *Form* tampilan riwayat hasil pelatihan

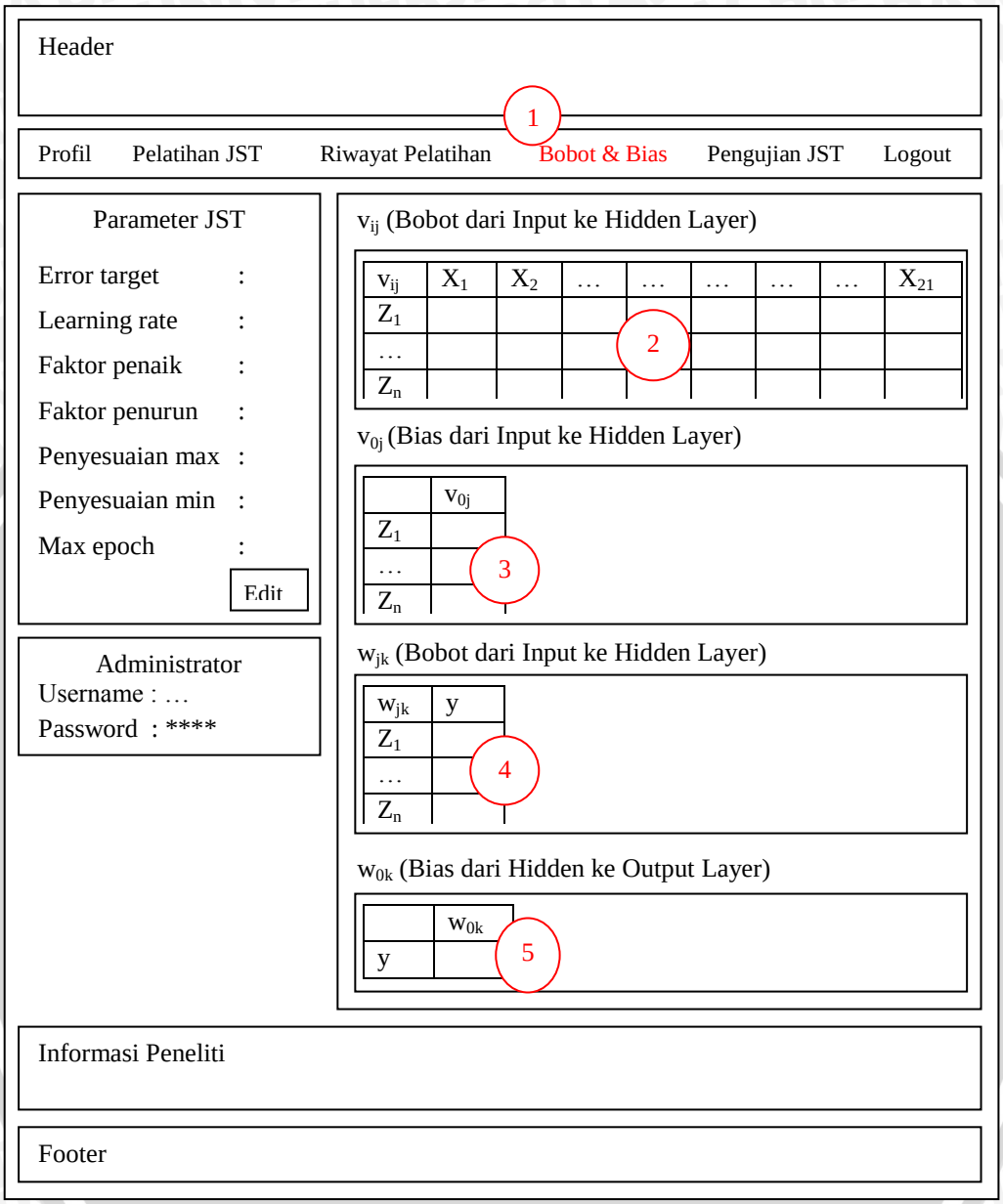


Gambar 3. 18 Rancangan *form* tampilan riwayat hasil pelatihan

Keterangan :

1. Menandakan bahwa pengguna sedang berada pada *form* tampilan riwayat hasil pelatihan setelah memilih menu "Riwayat Pelatihan".
2. Tabel untuk menyajikan data riwayat hasil pelatihan yang pernah dilakukan dengan bobot yang tersimpan. Data yang ditampilkan adalah jumlah input, *neuron hidden*, *error target*, *learning rate*, faktor penaik, faktor penurun, penyesuaian max, penyesuaian min, max *epoch*, jumlah data latih dan MSE.

c. *Form* tampilan data bobot dan bias terakhir



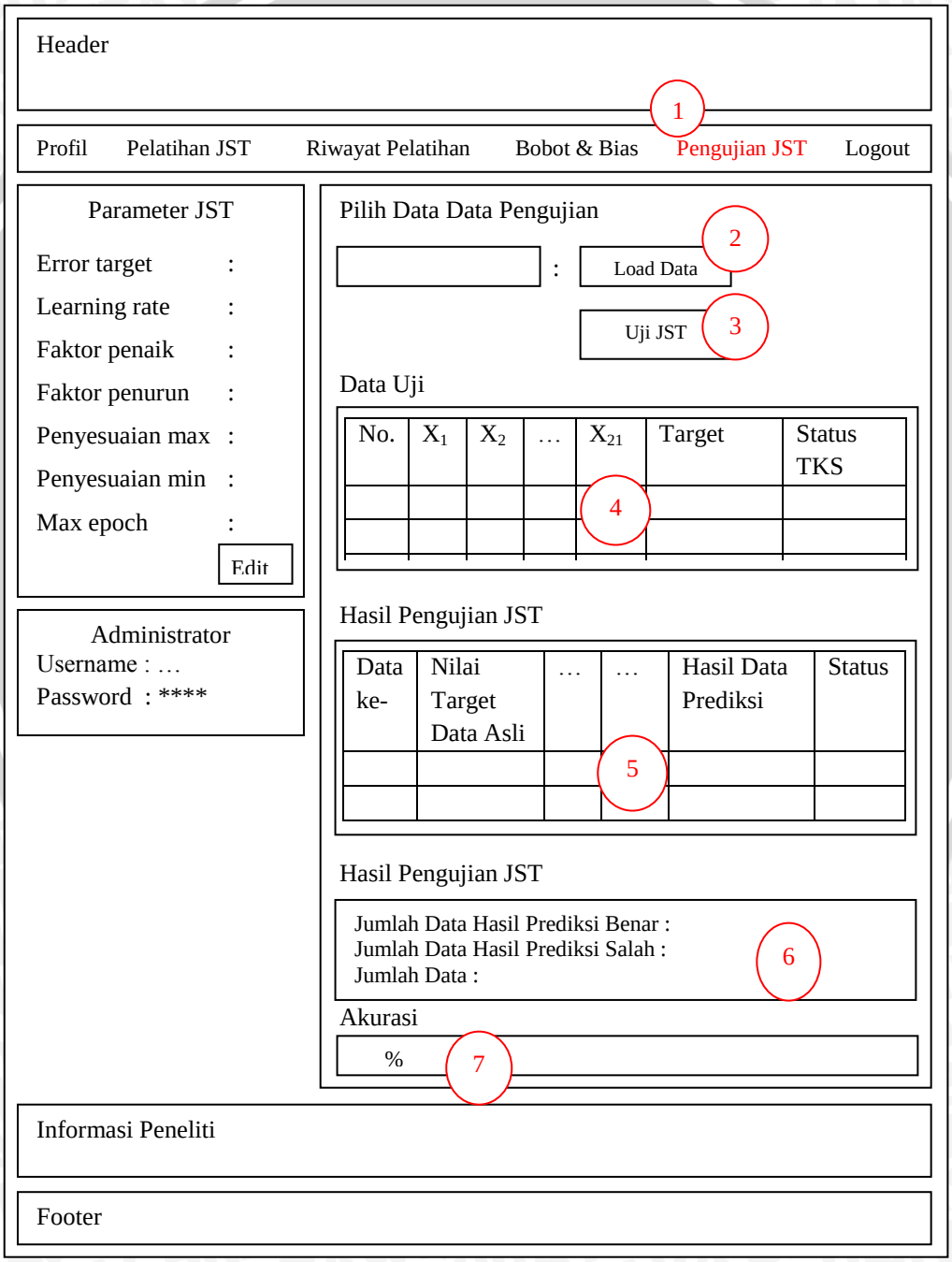
Gambar 3. 19 Rancangan *form* tampilan data bobot dan bias terakhir

Keterangan :

1. Menandakan bahwa pengguna sedang berada pada *form* tampilan data bobot dan bias terakhir setelah memilih menu “Bobot dan Bias”.
2. Tabel untuk menyajikan data berupa bobot dari input ke *hidden layer* (v_{ij}) hasil dari pelatihan JST yang disimpan pada dataset.
3. Tabel untuk menyajikan data berupa bias dari input ke *hidden layer* (v_{0j}) hasil dari pelatihan JST yang disimpan pada dataset.

4. Tabel untuk menyajikan data berupa bobot dari *hidden* ke *output layer* (w_{jk}) hasil dari pelatihan JST yang disimpan pada dataset.
5. Tabel untuk menyajikan data berupa bias dari *hidden* ke *output layer* (w_{jk}) hasil dari pelatihan JST yang disimpan pada dataset.

d. *Form* pengujian JST



Gambar 3. 20 Rancangan *form* pengujian JST

Keterangan :

1. Menandakan bahwa pengguna sedang berada pada *form* pengujian JST setelah memilih menu "Pengujian JST".
2. Tombol "*Load Data"* untuk memilih data yang akan diujikan dalam format csv.
3. Tombol "Uji JST" untuk memulai proses pengujian JST berdasarkan data yang telah di-*load*.
4. Tabel untuk menyajikan data pengujian yang telah di-*load*.
5. Tabel untuk manyajikan hasil pengujian JST berupa detail hasil perhitungan algoritma *feedforward* dan perbandingan hasil klasifikasi dengan data ali.
6. *Field* untuk menyajikan hasil pengujian JST berupa informasi jumlah data hasil prediksi benar, jumlah data hasil prediksi salah, dan jumlah data keseluruhan untuk pengujian.
7. *Field* untuk menyajikan akurasi hasil pengujian JST.

3.2.11 Perancangan Uji Coba

3.2.11.1 Rancangan Uji Coba Pengaruh Jumlah Neuron pada *Hidden Layer*

Uji coba ini dilakukan sebanyak 15 kali percobaan dengan menggunakan jumlah *hidden neuron* yang berbeda-beda pada masing-masing percobaan, yaitu 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75. Hal ini bertujuan untuk mengetahui pengaruh jumlah *hidden neuron* terhadap nilai MSE. Uji coba ini dirancang dengan menggunakan Tabel 3.49.

Tabel 3. 49 Rancangan uji coba pengaruh jumlah neuron pada *hidden layer*

Percobaan ke	Jumlah Neuron pada *Hidden Layer*	MSE
1	5	
2	10	
3	15	
4	20	
5	25	
…	…	
…	…	
15	75	

3.2.11.2 Rancangan Uji Coba Pengaruh Nilai *Learning Rate*

Uji coba ini dilakukan dengan menggunakan beberapa nilai *learning rate* yang diset berbeda-beda pada tiap percobaan. Hal ini bertujuan untuk mengetahui pengaruh nilai *learning rate* terhadap jumlah *epoch* yang dicapai hingga jaringan konvergen. Percobaan dilakukan sebanyak 10 kali dimulai dengan nilai *learning rate* terkecil sampai nilai *learning rate* terbesar, yaitu 0.1 sampai 1. Rancangan uji coba ini dapat dilihat pada Tabel 3.50.

Tabel 3. 50 Hasil percobaan pengaruh nilai *learning rate*

Percobaan ke	*Learning Rate*	Jumlah Epoch
1	0.1	
2	0.2	
3	0.3	
4	0.4	
…	…	
…	…	
10	1	

3.2.11.3 Hasil Pengujian

Pada langkah ini digunakan untuk mengetahui hasil uji coba nilai *learning rate* yang terbaik dan jumlah neuron pada *hidden layer* yang sesuai dengan konfigurasi jaringan yang digunakan. Konfigurasi jaringan optimal yang dihasilkan nantinya akan disajikan seperti pada Tabel 3.51.

Tabel 3. 51 Konfigurasi jaringan syaraf tiruan yang digunakan

Parameter	Nilai
Jumlah neuron pada *input layer*	
Jumlah neuron pada *hidden layer*	
Jumlah neuron pada *output layer*	
Error target	
Learning rate	
Faktor penaik	
Faktor penurun	
Penyesuaian maksimum	
Penyesuaian minimum	
Maksimum *epoch*	

Konfigurasi jaringan optimal yang telah dihasilkan akan digunakan untuk pengujian pada pengaruh jumlah data latih seimbang dan data latih tidak seimbang. Pengujian pada pengaruh jumlah data latih seimbang dan tidak seimbang ini digunakan untuk mengetahui keakuratan sistem dalam memberikan prediksi klasifikasi status tahapan keluarga sejahtera. Pengujian dilakukan sebanyak 7 kali dengan masing-masing pengujian terdapat 5 kali percobaan dengan proporsi data latih yang berbeda-beda. Variasi data latihnya, yaitu dikurangi setiap 10% data dari 70% data latih, sehingga secara berturut-turut data latih yang digunakan adalah sebanyak 20 data (10%), 40 data (20%), 60 data (30%), 80 data (40%), 100 data (50%), 120 data (60%), dan 140 data (70%), sedangkan data uji yang digunakan sebanyak 60 data (30%). Rancangan uji coba pengaruh data latih seimbang dapat dilihat pada Tabel 3.52 dan untuk data latih tidak seimbang dapat dilihat pada Tabel 3.53.

Tabel 3. 52 Hasil percobaan pengaruh data latih seimbang terhadap akurasi

Jumlah Data Latih (%)	Jumlah Data Latih	Jumlah Data Uji (30%)	Rata-rata Akurasi dari Percobaan 1 sampai 5	Akurasi Tertinggi dari Percobaan 1 sampai 5
10	20	60		
20	40	60		
30	60	60		
40	80	60		
50	100	60		
60	120	60		
70	140	60		

Tabel 3. 53 Hasil percobaan pengaruh data latih tidak seimbang terhadap akurasi

Jumlah Data Latih (%)	Jumlah Data Latih	Jumlah Data Uji (30%)	Rata-rata Akurasi dari Percobaan 1 sampai 5	Akurasi Tertinggi dari Percobaan 1 sampai 5
10	20	60		
20	40	60		
30	60	60		
40	80	60		
50	100	60		
60	120	60		
70	140	60		

BAB IV
IMPLEMENTASI

4.1 Lingkungan Implementasi

Implementasi merupakan tahapan penerapan dari rancangan sistem yang telah dibuat ke dalam bahasa pemrograman dan siap dioperasikan pada keadaan yang sebenarnya. Lingkungan implementasi yang akan dijelaskan pada sub bab ini adalah lingkungan perangkat keras dan lingkungan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan sistem klasifikasi status tahapan keluarga sejahtera dengan menggunakan metode pembelajaran *resilient backpropagation* ini sebagai berikut :

1. Processor Intel® Core™ i5 CPU M 460 @ 2.53GHz
2. RAM 2.00 GB
3. Harddisk dengan kapasitas 320 GB
4. Monitor 14.1”
5. Keyboard
6. Touchpad

4.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistem klasifikasi status tahapan keluarga sejahtera dengan menggunakan metode pembelajaran *resilient backpropagation* ini sebagai berikut :

1. Sistem Operasi Microsoft Windows 7 Ultimate 32-bit.
2. CodeLobster PHP Edition Free Version 4.2.1 sebagai *software development* dalam implementasi perancangan sistem.
3. Microsoft Excel 2010 dan MySQL 5.5.16 sebagai *software* pengolah database.

4.2 Implementasi Basis Data

Berdasarkan analisa dan perancangan yang terdapat pada bab 3 telah dijelaskan mengenai rancangan tabel dari basis data yang melibatkan 1 tabel saja.

Tabel tersebut adalah tabel Parameter. Tabel tersebut berfungsi untuk menyimpan data parameter JST yang akan digunakan dalam proses pelatihan dan pengujian. Tabel tersebut diimplementasikan dengan menggunakan MySQL 5.5.16, sedangkan penyimpanan data latih, data uji, dan data bobot hasil pelatihan diimplementasikan menggunakan Microsoft Excel 2010 dengan format penyimpanan CSV.

4.3 Implementasi Program

Penyusunan program untuk klasifikasi status tahapan keluarga sejahtera menggunakan metode pembelajaran *resilient backpropagation* memiliki beberapa bagian proses yang telah dirancang dan dijelaskan pada bab 3. Perangkat lunak yang dibangun adalah berupa aplikasi program berbasis web.

Implementasi dari proses-proses ini memanfaatkan struktur data berupa *array*, sehingga pada sub bab ini akan dijelaskan implementasi proses-proses tersebut.

4.3.1 Proses Inisialisasi Bobot dan Bias Awal

Pada proses inisialisasi bobot dan bias awal ini berguna untuk menginisialisasi nilai awal bobot dan bias dari unit-unit yang berada pada *input layer* ke unit-unit yang berada pada *hidden layer*, serta bobot dan bias dari unit-unit yang berada pada *hidden layer* ke unit-unit yang berada pada *output layer*.

Proses ini digunakan untuk menginisialisasi nilai bobot awal dari unit-unit yang berada pada *input layer* ke unit-unit *hidden layer* dan bobot dari unit-unit yang berada pada *hidden layer* ke unit-unit *output layer* serta unit-unit yang berada pada bias layer ke unit-unit *hidden layer* maupun ke unit-unit *output layer*. Bobot diinisialisasi secara acak dengan nilai [-0.05, 0.05], sedangkan untuk bobot pada *input layer* ke unit-unit *hidden layer* diinisialisasi dengan menggunakan algoritma Nguyen Widrow. Fungsinya ditunjukkan pada *Source code* 4.1.

```
function prosesNguyenWidrow($jumlah_input, $jumlah_hidden){
	$a = 1/$jumlah_input;
	$beta = 0.7 * pow($jumlah_hidden, $a);
	$minBeta = -1 * $beta;

	//inisialisasi bobot awal vij dengan random [-0.05, 0.05]
	for($j = 0; $j<$jumlah_hidden; $j++){
		for($i = 0; $i<$jumlah_input; $i++){
			$vij_awal[$i][$j]=rand(-5, 5);
			$vij[$i][$j]=0.01*$vij_awal[$i][$j];
		}
	}

	//menghitung nilai vj
	for($j = 0; $j<$jumlah_hidden; $j++){
		$vj[$j]=0;
		for($i = 0; $i<$jumlah_input; $i++){
			$vj[$j] +=($vij[$i][$j]*$vij[$i][$j]);
		}
		$vj[$j]=sqrt($vj[$j]);
	}

	//menginisialisasi bobot vij baru
	echo "<h3>1) Inisialisasi bobot vij</h3>";

	for($j = 0; $j<$jumlah_hidden; $j++){
		for($i = 0; $i<$jumlah_input; $i++){
			$vij_baru[$i][$j] = ($beta*$vij[$i][$j])/$vj[$j];
		}
	}

	//inisialisasi bobot awal v0j dengan random [-B, B]
	echo "<h3>2) Inisialisasi bias v0j</h3>";
	for($j = 0; $j<$jumlah_hidden; $j++){
		$v0j[$j] = float_random($beta, $minBeta, 4);
	}

	//inisialisasi bobot awal wjk dengan random [-0.05, 0.05]
	echo "<h3>3) Inisialisasi bobot wjk</h3>";
	for($j = 0; $j<$jumlah_hidden; $j++){
		$wjk_awal[$j] = rand(-5, 5);
		$wjk[$j]=0.01*$wjk_awal[$j];
	}

	//inisialisasi bobot awal w0k dengan random [-0.05, 0.05]
	echo "<h3>4) Inisialisasi bias w0k</h3>";
	$w0k_awal = rand(-5, 5);
	$w0k = 0.01*$w0k_awal;

	return array($vij_baru, $v0j, $wjk, $w0k);
}
```

Source code 4. 1 Proses Inisialisasi bobot dan bias awal

4.3.2 Proses *Feedforward*

Proses *feedforward* digunakan untuk mencari nilai output dari *neuron hidden* dan *neuron output*. Nilai tersebut kemudian akan diaktivasi menggunakan

fungsi *sigmoid biner*. Proses feedforward pada implementasi ini dibagi menjadi empat bagian, yaitu proses perhitungan nilai z_in_j, nilai z_j, nilai y_in_k, dan nilai y_k

Proses perhitungan nilai z_in_j digunakan untuk menampung sinyal masukan pada unit-unit yang berada pada *input layer* (x_i) yang nantinya akan dipropagasikan ke *hidden layer*. Proses perhitungan nilai z_j digunakan untuk meneruskan sinyal masukan pada unit-unit yang berada pada *input layer* yang telah ditampung pada proses z_in_j menuju ke *hidden layer* dengan menggunakan fungsi aktivasi *sigmoid biner*. Proses perhitungan nilai y_in_k digunakan untuk menampung keluaran dari setiap unit yang berada pada *hidden layer* yang nantinya akan dipropagasikan maju lagi ke *output layer*. Proses ini digunakan untuk meneruskan keluaran dari setiap unit yang berada pada *hidden layer* yang telah ditampung pada proses y_in_k menuju ke *output layer* dengan menggunakan fungsi aktivasi *sigmoid biner* hingga menghasilkan keluaran jaringan y_k. Fungsi *feedforward* ini ditunjukkan pada *Source code* 4.2.

```
function feedforward($xi, $vij, $v0j, $wjk, $w0k, $pola, $jumlah_input,
$jumlah_hidden){
	//Hitung nilai z_inj
	for($j = 0; $j<$jumlah_hidden; $j++){
		$sum_xivij[$j] = 0;
		for($i = 0; $i<$jumlah_input; $i++){
			$sum_xivij[$j]+=($xi[$pola][$i]*$vij[$i][$j]);
		}
		$z_inj[$j] = $v0j[$j]+$sum_xivij[$j];
	}
	//Hitung nilai zj
	for($j = 0; $j<$jumlah_hidden; $j++){
		$zj[$j] = 1/(1+exp(-1*($z_inj[$j])));
	}
	//Hitung nilai y_ink
	$sum_wjkzj = 0;
	for($j = 0; $j<$jumlah_hidden; $j++){
		$sum_wjkzj += ($wjk[$j]*$zj[$j]);
	}
	$y_ink = $w0k+$sum_wjkzj;
	//Hitung nilai yk
	$yk = 0;
	$yk = 1/(1+exp(-1*($y_ink)));

	return array($zj, $yk);
}
```

Source code 4. 2 Proses *feedforward*

4.3.3 Proses Info *Error Resilient Backpropagation*

Proses Perhitungan Informasi *Error resilient backpropagation* terdiri dari dua bagian, yaitu perhitungan nilai informasi *error* pada unit yang berada pada *output layer* dan nilai informasi *error* pada unit-unit yang berada pada *hidden layer*. Fungsinya ditunjukkan pada *Source code* 4.3.

```
function info_error($pola, $t, $yk, $zj, $wjk, $jumlah_hidden){
       $info_error_k = ($yk*(1-$yk))*0.5*($t[$pola]-$yk);

       for($j = 0; $j<$jumlah_hidden; $j++){
             $sum_error[$j] = 0;
             $sum_error[$j] += ($info_error_k*$wjk[$j]);
             $info_error_j[$j] = ($zj[$j]*(1-$zj[$j]))*$sum_error[$j];
       }

       return array($info_error_k, $info_error_j);
}
```

Source code 4. 3 Proses info *error resilient backpropagation*

4.3.4 Proses Hitung *Gradient*

Perhitungan nilai *gradient*, yaitu meliputi nilai *gradient* terhadap bobot pada unit-unit *hidden layer* ke unit *output layer*, nilai *gradient* terhadap bobot pada unit bias *layer* ke unit *output layer*, nilai *gradient* terhadap bobot pada unit-unit *input layer* ke unit-unit *hidden layer*, dan nilai *gradient* terhadap bobot pada unit bias *layer* ke unit-unit *hidden layer*. Fungsi hitung *gradient* ini ditunjukkan pada *Source code* 4.4.

```
function delta_E($delta_Evij_awal, $delta_Ev0j_awal, $delta_Ewjk_awal,
$delta_Ew0k_awal, $xi, $zj, $info_error_j, $info_error_k, $pola,
$jumlah_input, $jumlah_hidden){
    //Hitung gradient Ewjk
    for($j = 0; $j<$jumlah_hidden; $j++){
        $delta_Ewjk[$j] = $delta_Ewjk_awal[$j]+($info_error_k*$zj[$j]);
    }

    //Hitung gradient Ew0k
    $delta_Ew0k = 0;
    $delta_Ew0k = $delta_Ew0k_awal+($info_error_k*1);

    //Hitung gradient Evij
    for($j = 0; $j<$jumlah_hidden; $j++){
       for($i = 0; $i<$jumlah_input; $i++){
           $delta_Evij[$i][$j] = 0;
           $delta_Evij[$i][$j] = $delta_Evij_awal[$i][$j] +
                                 ($info_error_j[$j]*$xi[$pola][$i]);
       }
    }
```

```
    //Hitung gradient Ev0j
    for($j = 0; $j<$jumlah_hidden; $j++){
       $delta_Ev0j[$j] = 0;
       $delta_Ev0j[$j] = $delta_Ev0j_awal[$j]+($info_error_j[$j]*1);
    }

    return array($delta_Evij, $delta_Ev0j, $delta_Ewjk, $delta_Ew0k);
}
```

Source code 4. 4 Proses hitung *gradient*

4.3.5 Proses Perbaikan Bobot

Proses perbaikan bobot terdiri dari 4 proses, yaitu proses perbaikan bobot w_{jk}, perbaikan bobot w_{0k}, perbaikan bobot v_{ij}, dan perbaikan bobot v_{0j}.

a. Proses perbaikan bobot w_{jk}

Proses perbaikan bobot w_{jk} digunakan untuk menghitung bobot baru dari unit-unit pada *hidden layer* ke unit-unit pada *output layer*. Fungsinya ditunjukkan pada *Source code* 4.5.

```
function   perbaikiBobot_rprop_wjk($prev_learn_wjk,   $prev_delta_Ewjk,
$delta_Ewjk,  $peny_min,  $peny_max,  $faktor_penurun,  $faktor_penaik,
$wjk, $jumlah_hidden){
   for($j = 0; $j<$jumlah_hidden; $j++){
       $prevstep = max($prev_learn_wjk[$j], 0.00001);
       $grad_wjk = $delta_Ewjk[$j];
       $prev_grad_wjk = $prev_delta_Ewjk[$j];
       $grad_kali = sign($grad_wjk * $prev_grad_wjk);
       $sign_grad_wjk = sign($grad_wjk);
       if($grad_kali > 0){
              $nextstep = min($prevstep * $faktor_penaik, $peny_max);
       }
       else if($grad_kali < 0){
              $nextstep = max($prevstep * $faktor_penurun, $peny_min);
              $grad_wjk = 0;
       }
       else{
              $nextstep = $prevstep;
       }
       if($sign_grad_wjk > 0){
              $d_wjk[$j] = $nextstep;
       }
       else if($sign_grad_wjk < 0){
              $d_wjk[$j] = -1 * $nextstep;
       }
       else{
              $d_wjk[$j] = 0;
       }
       $wjk[$j] = $wjk[$j] + $d_wjk[$j];
       $prev_learn_wjk[$j] = $nextstep;
       $prev_delta_Ewjk[$j] = $grad_wjk;
    }
    return array($wjk, $prev_learn_wjk, $prev_delta_Ewjk);
}
```

Source code 4. 5 Proses perbaikan bobot w_{jk}

b. Proses perbaikan bobot w_{0k}

Proses perbaikan bobot w_{0k} digunakan untuk menghitung bobot baru dari unit-unit pada bias *layer* ke unit-unit pada *output layer*. Fungsinya ditunjukkan pada *Source code* 4.6.

```
function perbaikiBias_rprop_w0k($prev_learn_w0k, $prev_delta_Ew0k,
$delta_Ew0k, $peny_min, $peny_max, $faktor_penurun, $faktor_penaik,
$w0k){
	$prevstep = max($prev_learn_w0k, 0.00001);
	$grad_w0k = $delta_Ew0k;
	$prev_grad_w0k = $prev_delta_Ew0k;
	$grad_kali = sign($grad_w0k * $prev_grad_w0k);
	$sign_grad_w0k = sign($grad_w0k);
	if($grad_kali > 0){
		$nextstep = min($prevstep * $faktor_penaik, $peny_max);
	}
	else if($grad_kali < 0){
		$nextstep = max($prevstep * $faktor_penurun, $peny_min);
		$grad_w0k = 0;
	}
	else{
		$nextstep = $prevstep;
	}
	if($sign_grad_w0k > 0){
		$d_w0k = $nextstep;
	}
	else if($sign_grad_w0k < 0){
		$d_w0k = -1 * $nextstep;
	}
	else{
		$d_w0k = 0;
	}
	$w0k = $w0k + $d_w0k;
	$prev_learn_w0k = $nextstep;
	$prev_delta_Ew0k = $grad_w0k;
	return array($w0k, $prev_learn_w0k, $prev_delta_Ew0k);
}
```

Source code 4. 6 Proses perbaikan bobot w_{0k}

c. Proses perbaikan bobot v_{ij}

Proses perbaikan bobot v_{ij} digunakan untuk menghitung bobot baru dari unit-unit pada *input layer* ke unit-unit pada *hidden layer*. Fungsinya ditunjukkan pada *Source code* 4.7.

```
function perbaikiBobot_rprop_vij($prev_learn_vij, $prev_delta_Evij,
$delta_Evij, $peny_min, $peny_max, $faktor_penurun, $faktor_penaik,
$vij, $jumlah_input, $jumlah_hidden){
  for($j = 0; $j<$jumlah_hidden; $j++){
	for($i = 0; $i<$jumlah_input; $i++){
		$prevstep = max($prev_learn_vij[$i][$j], 0.00001);
		$grad_vij = $delta_Evij[$i][$j];
		$prev_grad_vij = $prev_delta_Evij[$i][$j];
		$grad_kali = sign($grad_vij * $prev_grad_vij);
```

```
                $sign_grad_vij = sign($grad_vij);

                if($grad_kali > 0){
                        $nextstep = min($prevstep * $faktor_penaik,
                                $peny_max);
                }
                else if($grad_kali < 0){
                        $nextstep = max($prevstep * $faktor_penurun,
                                $peny_min);
                        $grad_vij = 0;
                }
                else{
                        $nextstep = $prevstep;
                }

                if($sign_grad_vij > 0){
                        $d_vij[$i][$j] = $nextstep;
                }
                else if($sign_grad_vij < 0){
                        $d_vij[$i][$j] = -1 * $nextstep;
                }
                else{
                        $d_vij[$i][$j] = 0;
                }

                $vij[$i][$j] = $vij[$i][$j] + $d_vij[$i][$j];
                $prev_learn_vij[$i][$j] = $nextstep;
                $prev_delta_Evij[$i][$j] = $grad_vij;
        }
    }

        return array($vij, $prev_learn_vij, $prev_delta_Evij);
}
```

Source code 4. 7 Proses perbaikan bobot v_{ij}

d. Proses perbaikan bobot v_{0j}

Proses perbaikan bobot v_{0j} digunakan untuk menghitung bobot baru dari unit-unit pada bias *layer* ke unit-unit pada *hidden layer*. Fungsinya ditunjukkan pada *Source code* 4.8.

```
function perbaikiBias_rprop_v0j($prev_learn_v0j, $prev_delta_Ev0j,
$delta_Ev0j, $peny_min, $peny_max, $faktor_penurun, $faktor_penaik,
$v0j, $jumlah_hidden){
        for($j = 0; $j<$jumlah_hidden; $j++){
                $prevstep = max($prev_learn_v0j[$j], 0.00001);
                $grad_v0j = $delta_Ev0j[$j];
                $prev_grad_v0j = $prev_delta_Ev0j[$j];
                $grad_kali = sign($grad_v0j * $prev_grad_v0j);
                $sign_grad_v0j = sign($grad_v0j);

                if($grad_kali > 0){
                        $nextstep = min($prevstep * $faktor_penaik,
                                $peny_max);
                }
                else if($grad_kali < 0){
                        $nextstep = max($prevstep * $faktor_penurun,
```

```
                                $peny_min);
                        $grad_v0j = 0;
                }
                else{
                        $nextstep = $prevstep;
                }

                if($sign_grad_v0j > 0){
                        $d_v0j[$j] = $nextstep;
                }
                else if(($sign_grad_v0j < 0)){
                        $d_v0j[$j] = -1 * $nextstep;
                }
                else{
                        $d_v0j[$j] = 0;
                }
                $v0j[$j] = $v0j[$j] + $d_v0j[$j];
                $prev_learn_v0j[$j] = $nextstep;
                $prev_delta_Ev0j[$j] = $grad_v0j;
        }
        return array($v0j, $prev_learn_v0j, $prev_delta_Ev0j);
}
```

Source code 4. 8 Proses perbaikan bobot v_{0j}

4.3.6 Proses Pelatihan JST *Resilient Backpropagation*

Dalam proses pelatihan terdapat beberapa fungsi yang dijalankan, yaitu proses inisialisasi bobot dan bias awal dengan algortima Nguyen Widrow, *feedfoward*, info *error resillientpropagation*, hitung *gradient*, perbaikan bobot w_{jk}, perbaikan bobot w_{0k}, perbaikan bobot v_{ij}, perbaikan bobot v_{0j}, dan perhitungan MSE (*Mean Square Error*). Implementasi proses pelatihan dapat dilihat pada *Source code* 4.9.

```
function pelatihan($data_latih, $jumlah_hidden, $learn_rate,
$target_error, $max_epoch, $peny_min, $peny_max, $faktor_penurun,
$faktor_penaik){
        $status = false;

        echo "<h3>Data Input</h3>";
        echo "<div style='border: 1px solid rgb(204, 204, 204);
              padding: 5px; overflow: auto; width: 671px; height:
              380px; background-color: rgb(255, 255, 255)'>";
        $xi = get2DArray_Input($data_latih, ",");
        echo "</div><br>";

        $t = get1DArray_Target($data_latih, ",");
        $jumlah_data = jumlahData($data_latih, ",");
        echo "<h2>Inisialisasi Bobot dan Bias Awal</h2>";
        echo "<div style='border: 1px solid rgb(204, 204, 204);
              padding: 5px; overflow: auto; width: 671px; height:
              380px; background-color: rgb(255, 255, 255)'>";
        list($vij_awal, $v0j_awal, $wjk_awal, $w0k_awal) =
       prosesNguyenWidrow(21, $jumlah_hidden);
        echo "</div><br>";
```

```
$delta_Evij_awal = delta_Evij_awal(21, $jumlah_hidden);
$delta_Ev0j_awal = delta_Ev0j_awal($jumlah_hidden);
$delta_Ewjk_awal = delta_Ewjk_awal($jumlah_hidden);
$delta_Ew0k_awal = delta_Ew0k_awal();

$learnR_wjk_awal = learnR_wjk($learn_rate,
                   $jumlah_hidden);
$learnR_w0k_awal = learnR_w0k($learn_rate);
$learnR_vij_awal = learnR_vij($learn_rate, 21,
                   $jumlah_hidden);
$learnR_v0j_awal = learnR_v0j($learn_rate,
                   $jumlah_hidden);

$mse_awal = mse_awal(0);

$epoch = 0;
while($status == false){
      $pola = 0;
      while($pola < $jumlah_data){
         if($epoch < 1){
           //Feedforward
           list($zj, $yk) = feedforward($xi, $vij_awal,
         $v0j_awal, $wjk_awal, $w0k_awal, $pola, 21,
         $jumlah_hidden);

           //RPROP
           list($info_error_k, $info_error_j) =
         info_error($pola, $t, $yk, $zj, $wjk_awal,
         $jumlah_hidden);

           if($pola < 1){
                list($delta_Evij, $delta_Ev0j,
                $delta_Ewjk, $delta_Ew0k) =
                delta_E($delta_Evij_awal,
                $delta_Ev0j_awal, $delta_Ewjk_awal,
                $delta_Ew0k_awal, $xi, $zj,
                $info_error_j, $info_error_k, $pola, 21,
                $jumlah_hidden);

                $mse_temp = mse_temp($pola, $mse_awal,
                            $t, $yk);
           }

           else{
               list($delta_Evij, $delta_Ev0j,
               $delta_Ewjk, $delta_Ew0k) =
               delta_E($delta_Evij, $delta_Ev0j,
               $delta_Ewjk, $delta_Ew0k, $xi, $zj,
               $info_error_j, $info_error_k, $pola, 21,
               $jumlah_hidden);

               $mse_temp = mse_temp($pola, $mse_temp,
                           $t, $yk);
           }
         }

        else if($epoch >= 1){
           //Feedforward
           list($zj, $yk) = feedforward($xi,
           $bobot_baru_vij, $bias_baru_v0j,
           $bobot_baru_wjk, $bias_baru_w0k, $pola, 21, \
           $jumlah_hidden);

           //RPROP
```

```
            list($info_error_k, $info_error_j) =
            info_error($pola, $t, $yk, $zj,
            $bobot_baru_wjk, $jumlah_hidden);

            if($pola < 1){
                  list($delta_Evij, $delta_Ev0j,
                  $delta_Ewjk, $delta_Ew0k) =
                  delta_E($delta_Evij_awal,
                  $delta_Ev0j_awal, $delta_Ewjk_awal,
                  $delta_Ew0k_awal, $xi, $zj,
                  $info_error_j, $info_error_k, $pola, 21,
                  $jumlah_hidden);

                  $mse_temp = mse_temp($pola, $mse_awal,
                              $t, $yk);
            }
            else{
                  list($delta_Evij, $delta_Ev0j,
                  $delta_Ewjk, $delta_Ew0k) =
                  delta_E($delta_Evij, $delta_Ev0j,
                  $delta_Ewjk, $delta_Ew0k, $xi, $zj,
                  $info_error_j, $info_error_k, $pola, 21,
                  $jumlah_hidden);

                  $mse_temp = mse_temp($pola, $mse_temp,
                              $t, $yk);
            }
      }
     $pola++;
     }

     $mse_akhir = mse($mse_temp, $data_latih, $epoch);
     $mse_array2[$epoch] = mse_array($mse_akhir, $epoch);
     echo $mse_array3[$epoch] = $mse_array2[$epoch].
    "<br>";

     if($epoch < 1){
            list($bobot_baru_wjk, $learnR_baru_wjk,
            $prev_delta_Ewjk) =
            perbaikiBobot_rprop_wjk($learnR_wjk_awal,
            $delta_Ewjk_awal, $delta_Ewjk, $peny_min,
            $peny_max, $faktor_penurun, $faktor_penaik,
            $wjk_awal, $jumlah_hidden);

            list($bias_baru_w0k, $learnR_baru_w0k,
            $prev_delta_Ew0k) =
            perbaikiBias_rprop_w0k($learnR_w0k_awal,
            $delta_Ew0k_awal, $delta_Ew0k, $peny_min,
            $peny_max, $faktor_penurun, $faktor_penaik,
            $w0k_awal);

            list($bobot_baru_vij, $learnR_baru_vij,
            $prev_delta_Evij) =
            perbaikiBobot_rprop_vij($learnR_vij_awal,
            $delta_Evij_awal, $delta_Evij, $peny_min,
            $peny_max, $faktor_penurun, $faktor_penaik,
            $vij_awal, 21, $jumlah_hidden);

            list($bias_baru_v0j, $learnR_baru_v0j,
            $prev_delta_Ev0j) =
            perbaikiBias_rprop_v0j($learnR_v0j_awal,
            $delta_Ev0j_awal, $delta_Ev0j, $peny_min,
            $peny_max, $faktor_penurun, $faktor_penaik,
            $v0j_awal, $jumlah_hidden);
```

```
        }
        else if($epoch >= 1){
                list($bobot_baru_wjk, $learnR_baru_wjk,
                $prev_delta_Ewjk) =
                perbaikiBobot_rprop_wjk($learnR_baru_wjk,
                $prev_delta_Ewjk, $delta_Ewjk, $peny_min,
                $peny_max, $faktor_penurun, $faktor_penaik,
                $bobot_baru_wjk, $jumlah_hidden);

                list($bias_baru_w0k, $learnR_baru_w0k,
                $prev_delta_Ew0k) =
                perbaikiBias_rprop_w0k($learnR_baru_w0k,
                $prev_delta_Ew0k, $delta_Ew0k, $peny_min,
                $peny_max, $faktor_penurun, $faktor_penaik,
                $bias_baru_w0k);

                list($bobot_baru_vij, $learnR_baru_vij,
                $prev_delta_Evij) =
                perbaikiBobot_rprop_vij($learnR_baru_vij,
                $prev_delta_Evij, $delta_Evij, $peny_min,
                $peny_max, $faktor_penurun, $faktor_penaik,
                $bobot_baru_vij, 21, $jumlah_hidden);

                list($bias_baru_v0j, $learnR_baru_v0j,
                $prev_delta_Ev0j) =
                perbaikiBias_rprop_v0j($learnR_baru_v0j,
                $prev_delta_Ev0j, $delta_Ev0j, $peny_min,
                $peny_max, $faktor_penurun, $faktor_penaik,
                $bias_baru_v0j, $jumlah_hidden);
        }
        if($mse_akhir < $target_error){
                echo "<h2>MSE</h2>";
                echo "<div style='border: 1px solid rgb(204,
                    204, 204); padding: 5px; overflow: auto;
                    width: 671px; height: 105px; background-
                    color: rgb(255, 255, 255)'>";
                $mse_akhir = mse($mse_temp, $data_latih,
                                $epoch);
                $status = true;
                printMSE($mse_akhir, $epoch);
                echo "<br>";
                echo "<b>Epoch ke-".($epoch+1)."</b><br>";
                echo "<b>MSE STOP</b><br>";
                echo "</div><br>";
                echo "<h2>Bobot dan Bias Akhir Hasil Pelatihan
                    JST</h2>";
                echo "<div style='border: 1px solid rgb(204,
                    204, 204); padding: 5px; overflow: auto;
                    width: 671px; height: 380px; background-
                    color: rgb(255, 255, 255)'>";
                printBobot_vij($bobot_baru_vij, 21,
                                $jumlah_hidden, $epoch);
                printBias_v0j($bias_baru_v0j, $jumlah_hidden);
                printBobot_wjk($bobot_baru_wjk,
                                $jumlah_hidden);
                printBias_w0k($bias_baru_w0k);
                echo "</div><br>";
                simpanBobot($bobot_baru_vij, $bias_baru_v0j,
                $bobot_baru_wjk, $bias_baru_w0k, 21,
                $jumlah_hidden);
                simpanDataPelatihan(21, $jumlah_hidden,
              $learn_rate, $target_error, $max_epoch,
              $peny_min, $peny_max, $faktor_penurun,
              $faktor_penaik, $jumlah_data, $mse_akhir);
```

```
                    ?>
                    <form method="POST"
                     action="m_simpan_bobot.php"
                     enctype="multipart/form-data">
                       <br />
                       <input type="submit" name="SimpanBobot"
                        value="Simpan Bobot" style="border:1px
                        solid; padding:5px 20px 5px 10px;
                        margin-top:3px; margin-left:565px;"/>
                    </form>
                    <?php
                }
                if($epoch == $max_epoch-1){
                    echo "<h2>MSE</h2>";
                    echo "<div style='border: 1px solid rgb(204,
                        204, 204); padding: 5px; overflow: auto;
                        width: 671px; height: 105px; background-
                        color: rgb(255, 255, 255)'>";
                    $mse_akhir = mse($mse_temp, $data_latih,
                                $epoch);
                    $status = true;
                    printMSE($mse_akhir, $epoch);
                    echo "<br>";
                    echo "<b>Epoch ke-".($epoch+1)."</b><br>";
                    echo "<b>EPOCH STOP</b><br>";
                    echo "</div><br>";
                    echo "<h2>Bobot dan Bias Akhir Hasil Pelatihan
                        JST</h2>";
                    echo "<div style='border: 1px solid rgb(204,
                        204, 204); padding: 5px; overflow: auto;
                        width: 671px; height: 380px; background-
                        color: rgb(255, 255, 255)'>";
                    printBobot_vij($bobot_baru_vij, 21,
                                $jumlah_hidden, $epoch);
                    printBias_v0j($bias_baru_v0j, $jumlah_hidden);
                    printBobot_wjk($bobot_baru_wjk,
                                $jumlah_hidden);
                    printBias_w0k($bias_baru_w0k);
                    echo "</div><br>";
                    simpanBobot($bobot_baru_vij, $bias_baru_v0j,
                    $bobot_baru_wjk, $bias_baru_w0k, 21,
                    $jumlah_hidden);
                    simpanDataPelatihan(21, $jumlah_hidden,
                    $learn_rate, $target_error, $max_epoch,
                    $peny_min, $peny_max, $faktor_penurun,
                    $faktor_penaik, $jumlah_data, $mse_akhir);
                    ?>
                    <form method="POST"
                      action="m_simpan_bobot.php"
                      enctype="multipart/form-data">
                        <br />
                        <input type="submit" name="SimpanBobot"
                         value="Simpan Bobot" style="border:1px
                         solid; padding:5px 20px 5px 10px;
                         margin-top:3px; margin-left:565px;"/>
                    </form>
                    <?php
                }
        $epoch++;
        }
}
```

Source code 4. 9 Proses pelatihan JST *resilient backpropagation*

4.3.7 Proses Pengujian JST

Dalam proses pengujian nilai bobot optimal yang dihasilkan pada tahap pelatihan digunakan untuk inisialisasi bobot awal digunakan sebagai pengujian klasifikasi data status tahapan keluarga sejahtera. Proses pengujian JST ini dilakukan dengan menjalankan proses *feedforward* setelah bobot awal terinisialisasi. Implementasi proses pengujian dapat dilihat pada *Source code* 4.10.

```
function pengujian($data_uji, $jumlah_hidden){
        echo "<h2>Data Uji</h2>";
        echo "<div style='border: 1px solid rgb(204, 204, 204);
              padding: 5px; overflow: auto; width: 671px; height:
              380px; background-color: rgb(255, 255, 255)'>";
        $xi = get2DArray_Input($data_uji, ",");
        $t = get1DArray_Target($data_uji,',');
        $jumlah_data = jumlahData($data_uji, ",");
        echo "</div><br>";
        $vij = get2DArray('data/bobot_vij.csv',',');
        $v0j = get1DArray('data/bias_v0j.csv',',');
        $wjk = get1DArray('data/bobot_wjk.csv',',');
        $w0k = getBias_w0k('data/bias_w0k.csv',',');

        $yk = feedforward($xi, $vij, $v0j, $wjk, $w0k,
              $jumlah_hidden, $jumlah_data);
        prediksi($jumlah_data, $t, $yk);
}
```

Source code 4. 10 Proses pengujian JST

4.3.8 Proses Prediksi

Pada proses prediksi dilakukan pencocokan antara nilai keluaran hasil pengujian JST *feedforward* dengan nilai target data asli. Apabila nilai keluaran JST masih dalam rentang nilai target, maka akan dianggap sesuai dengan nilai target data asli. Selanjutnya dihitung jumlah hasil klasifikasi benar dan salah, serta dihitung akurasi prediksi sistem dengan menggunakan proporsi data latih tertentu. Implementasi proses prediksi dapat dilihat pada *Source code* 4.11.

```
function prediksi($jumlah_data, $t, $yk){
        $batas1 = 0.0;
        $batas2 = 0.2;
        $batas3 = 0.4;
        $batas4 = 0.6;
        $batas5 = 0.8;
        $batas6 = 1.0;

        echo "<h2>Hasil Pengujian JST</h2>";
        echo "<div style='border: 1px solid rgb(204, 204, 204); padding:
5px; overflow: auto; width: 671px; height: 380px; background-color:
rgb(255, 255, 255)'>";
```

```
echo "<table class='table-list' style='border:1px solid'
     width='670'>";
echo "<tr>";
echo "<td align='center' width='12%'
     bgcolor='#CCCCCC'><b>Data ke-</b></td>";
echo "<td align='center' width='19%'
     bgcolor='#CCCCCC'><b>Nilai Target Data
     Asli</b></td>";
echo "<td align='center' width='17%'
     bgcolor='#CCCCCC'><b>Target Data Asli</b></td>";
echo "<td align='center' width='22%'
     bgcolor='#CCCCCC'><b>Nilai Hasil Data
     Prediksi</b></td>";
echo "<td align='center' width='19%'
     bgcolor='#CCCCCC'><b>Hasil Data Prediksi</b></td>";
echo "<td align='center'
     bgcolor='#CCCCCC'><b>Status</b></td>";
echo "</tr>";
for($i = 0; $i<$jumlah_data; $i++){
       $nilai_target[$i] = $t[$i];
       $nilai_yk[$i] = $yk[$i];
       if($nilai_target[$i] >= $batas1 && $nilai_target[$i]
      < $batas2){
              $target[$i] = "Prasejahtera";
       }
       else if($nilai_target[$i] >= $batas2 &&
      $nilai_target[$i] < $batas3){
              $target[$i] = "KS I";
       }
       else if($nilai_target[$i] >= $batas3 &&
      $nilai_target[$i] < $batas4){
              $target[$i] = "KS II";
       }
       else if($nilai_target[$i] >= $batas4 &&
      $nilai_target[$i] < $batas5){
              $target[$i] = "KS III";
       }
       else if($nilai_target[$i] >= $batas5 &&
      $nilai_target[$i] < $batas6){
              $target[$i] = "KS III+";
       }
       else{
              $target[$i] = "Tidak Ada";
       }

       if($nilai_yk[$i] >= $batas1 && $nilai_yk[$i] <
      $batas2){
              $yk_ks[$i] = "Prasejahtera";
       }
       else if($nilai_yk[$i] >= $batas2 && $nilai_yk[$i] <
      $batas3){
              $yk_ks[$i] = "KS I";
       }
       else if($nilai_yk[$i] >= $batas3 && $nilai_yk[$i] <
      $batas4){
              $yk_ks[$i] = "KS II";
       }
       else if($nilai_yk[$i] >= $batas4 && $nilai_yk[$i] <
      $batas5){
              $yk_ks[$i] = "KS III";
       }
       else if($nilai_yk[$i] >= $batas5 && $nilai_yk[$i] <
      $batas6){
              $yk_ks[$i] = "KS III+";
```

```
            }
            else{
                  $yk_ks[$i] = "Tidak Ada";
            }

            if($yk_ks[$i] == $target[$i]){
                  $status[$i] = "Benar";
                  $hasil_benar[$i] = $status[$i];
                  $jumlah_benar = count($hasil_benar);
                  if($jumlah_benar == $jumlah_data){
                        $jumlah_salah = 0;
                  }
            }
            else{
                  $status[$i] = "Salah";
                  $hasil_salah[$i] = $status[$i];
                  $jumlah_salah = count($hasil_salah);
                  if($jumlah_salah == $jumlah_data){
                        $jumlah_benar = 0;
                  }
            }
            echo "<tr>";
            echo "<td align='center'>".($i+1)."</td>";
            echo "<td
                 align='center'>".$nilai_target[$i]."</td>";
            echo "<td align='center'>".$target[$i]."</td>";
            echo "<td align='center'>".$nilai_yk[$i]."</td>";
            echo "<td align='center'>".$yk_ks[$i]."</td>";
            if($status[$i] == "Salah"){
                  echo "<td align='center'
                 style='color:#f20000';>".$status[$i]."</td>";
            }
            else{
                  echo "<td align='center'
                 style='color:#3456cb';>".$status[$i]."</td>";
            }
            echo "</tr>";
      }
      echo "</table>";
      echo "</div><br>";

      echo "<h2>Hasil Pengujian JST</h2>";
      echo "<div style='border: 1px solid rgb(204, 204, 204);
           padding: 5px; overflow: auto; width: 671px; height:
           80px; background-color: rgb(255, 255, 255)'>";
      echo "Jumlah Data Hasil Prediksi Benar :
          ".$jumlah_benar."<br>";
      echo "Jumlah Data Hasil Prediksi Salah :
          ".$jumlah_salah."<br>";
      echo "Jumlah Data : ".$jumlah_data."<br>";
      echo "</div><br>";

      echo "<h2>Akurasi</h2>";
      echo "<div style='border: 1px solid rgb(204, 204, 204);
           padding: 5px; overflow: auto; width: 671px; height:
           20px; background-color: rgb(255, 255, 255)'>";
      $akurasi = round($jumlah_benar/$jumlah_data * 100, 3);
      echo "<b>".$akurasi."%</b>";
      echo "</div><br>";
}
```

Source code 4. 11 Proses prediksi

4.4 Implementasi Antarmuka

Berdasarkan rancangan antarmuka pada sub bab 3.2.9 maka dihasilkan antarmuka yang terdiri dari antarmuka utama dan antarmuka administrator.

4.4.1 Antarmuka Utama

Antarmuka utama terdiri dari *form* utama yang berisi menu Home, Deskripsi Sistem, Tentang JST RPOP, dan Aplikasi. Halaman utama dapat dilihat pada Gambar 4.1.

Gambar 4. 1 Halaman utama

4.4.2 Antarmuka Administrator

Antarmuka administrator terdiri dari *form* utama yang berisi menu Profil, Pelatihan JST, Riwayat Pelatihan, Bobot & Bias, Pengujian JST, dan Logout. Antarmuka administrator dapat dilihat pada Gambar 4.2 sampai Gambar 4.7.

a. Form Profil

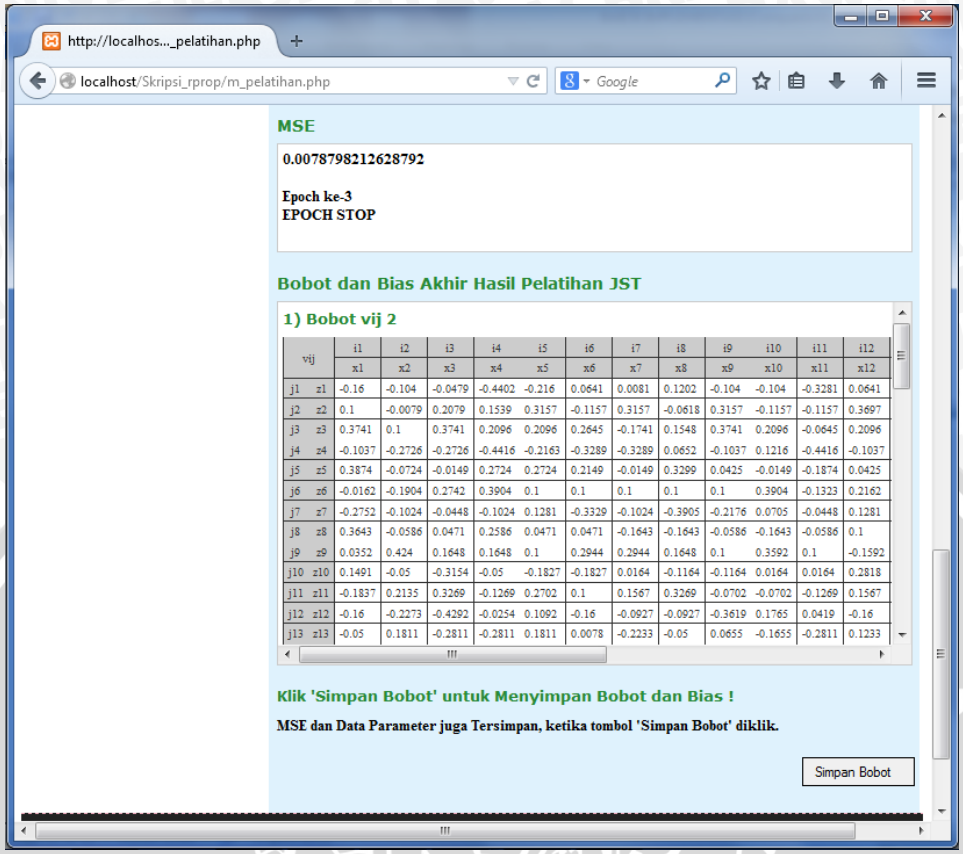
Form profil ini berisi mengenai informasi data detail administrator yang sedang mengakses halaman admin. *Form* profil dapat dilihat pada Gambar 4.2.

Gambar 4. 2 *Form* profil

b. Form Pelatihan JST

Pada *form* pelatihan JST, admin dapat melakukan proses pelatihan pada sistem dengan menentukan parameter-parameter jaringan yang terdiri dari jumlah neuron pada *hidden layer*, *error target*, *learning rate*, dan max *epoch*. Apabila parameter sudah diatur, admin dapat membuka file data input status tahapan keluarga sejahtera yang akan dilatihkan, kemudian menekan tombol "Load Data" untuk proses menampilkan data ke dalam *form* pelatihan. Setelah data ditampilkan, maka admin harus memilih data input yang sama dengan cara klik "*Choose File"* dan kemudian menekan tombol "Latih JST" untuk melakukan proses pelatihan JST *resilient backpropagation*. Setelah proses pelatihan selesai, maka akan ditampilkan nilai MSE beserta epoch yang dicapai dan nilai bobot optimal yang akan disimpan. Bobot hasil pelatihan dapat disimpan dengan menekan tombol "Simpan Bobot", maka riwayat pelatihan juga akan ikut tersimpan. *Form* pelatihan JST dapat dilihat pada Gambar 4.3 dan *form* hasil pelatihan JST dapat dilihat pada Gambar 4.4.

Gambar 4. 3 *Form* pelatihan JST



Gambar 4. 4 *Form* hasil pelatihan JST

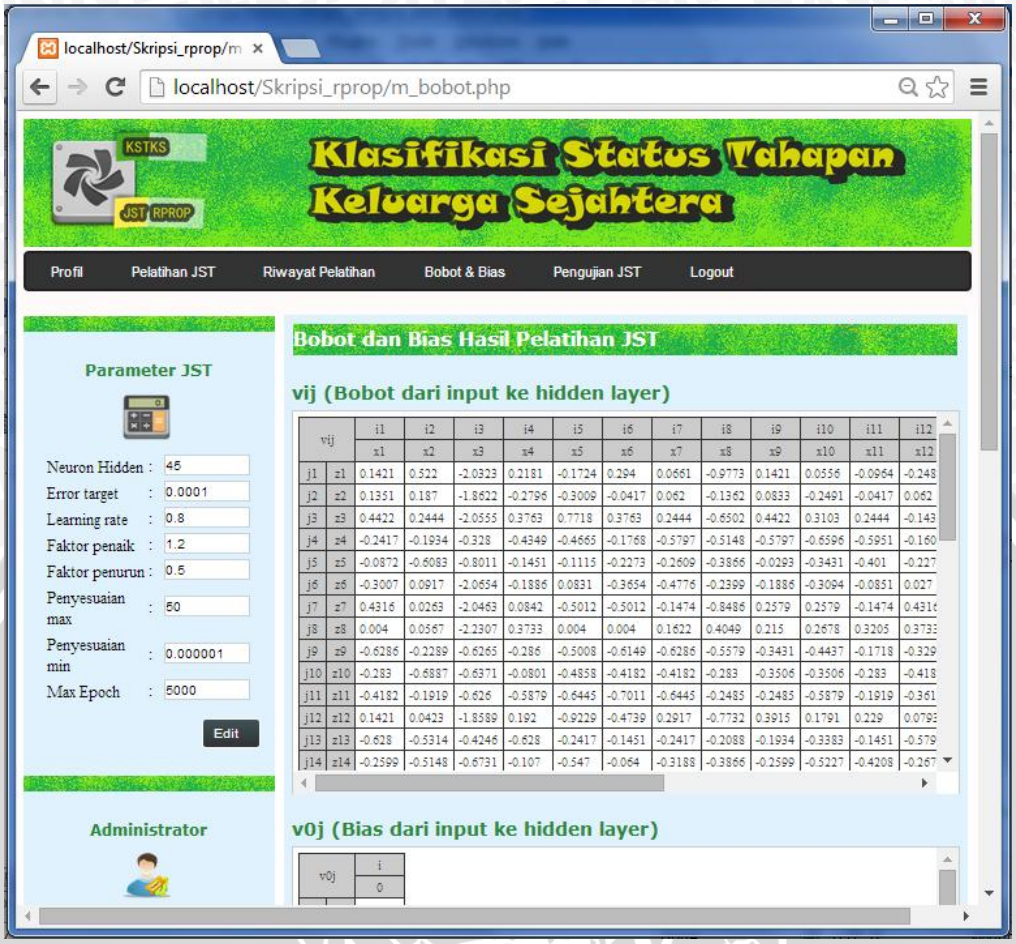
c. *Form* Riwayat Pelatihan

Pada *form* riwayat pelatihan, admin dapat mengetahui nilai parameter yang pernah dipakai dalam percobaan dan menampilkan nilai MSE percobaan pelatihan JST yang pernah dilakukan. Selain itu, admin juga bisa menghapus data riwayat pelatihan dengan menekan tombol "Delete". *Form* riwayat pelatihan dapat dilihat pada Gambar 4.5.

Gambar 4. 5 *Form* riwayat pelatihan

d. Form Bobot dan Bias

Pada *form* bobot dan bias, admin dapat mengetahui nilai bobot dan bias hasil pelatihan yang telah disimpan. *Form* bobot dan bias dapat dilihat pada Gambar 4.6.

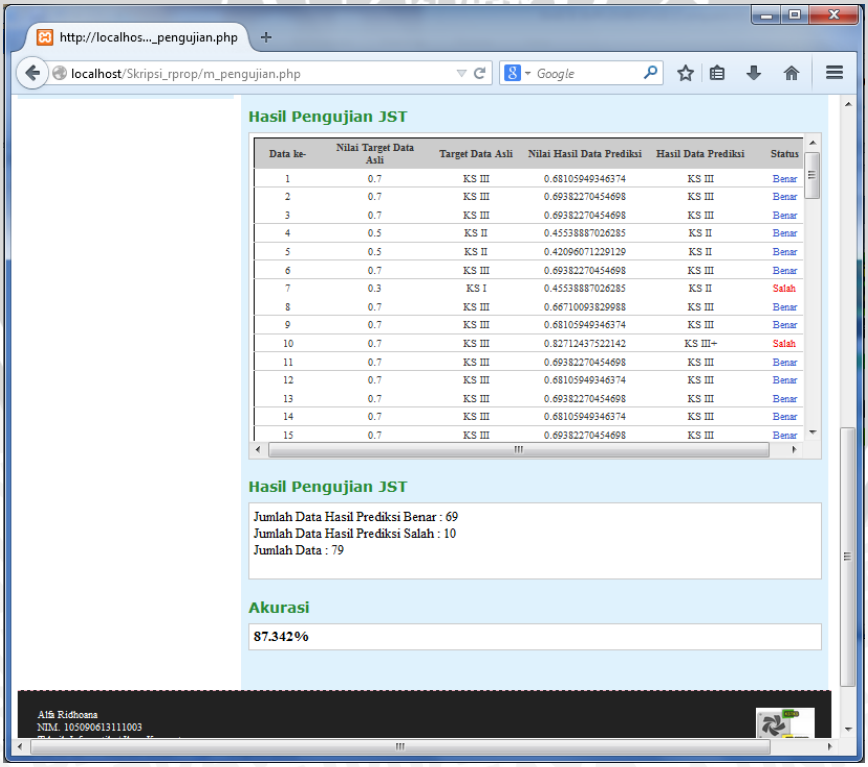


Gambar 4. 6 *Form* bobot dan bias

e. Form Pengujian JST

Pada *form* pengujian JST, admin dapat melakukan proses pengujian pada sistem. Parameter jumlah neuron pada *hidden layer* harus sama seperti pada saat pelatihan. Untuk memulainya, admin dapat membuka file data input status tahapan keluarga sejahtera yang akan diujikan, kemudian menekan tombol "Load Data" untuk proses menampilkan data ke dalam *form* pengujian. Setelah data ditampilkan, maka admin harus memilih data input yang sama dengan cara klik "*Choose File"* dan kemudian menekan tombol "Uji JST" untuk melakukan proses pengujian JST dengan algoritma *feedforward*. Setelah proses pengujian selesai, maka akan ditampilkan nilai keluaran hasil *feedforwad* dan nilai target pada data asli serta akurasi prediksi sistem. *Form* pengujian JST dapat dilihat pada Gambar 4.7 dan *form* hasil pengujian JST dapat dilihat pada Gambar 4.8.

Gambar 4. 7 *Form* pengujian JST



Gambar 4. 8 *Form* hasil pengujian JST

BAB V
PENGUJIAN DAN ANALISIS

5.1 Uji Coba Sistem

Untuk memperoleh konfigurasi jaringan syaraf tiruan terbaik yang digunakan untuk mengklasifikasikan status tahapan keluarga sejahtera, maka dilakukan uji coba terhadap sistem. Uji coba dilakukan dengan cara melakukan pelatihan pada jaringan syaraf tiruan dengan menggunakan parameter-parameter yang telah ditentukan. Uji coba dilakukan guna menentukan jumlah *hidden neuron* dan nilai *learning rate* yang optimal, sehingga dapat diambil konfigurasi jaringan yang terbaik untuk digunakan dalam mengklasifikasikan status tahapan keluarga sejahtera.

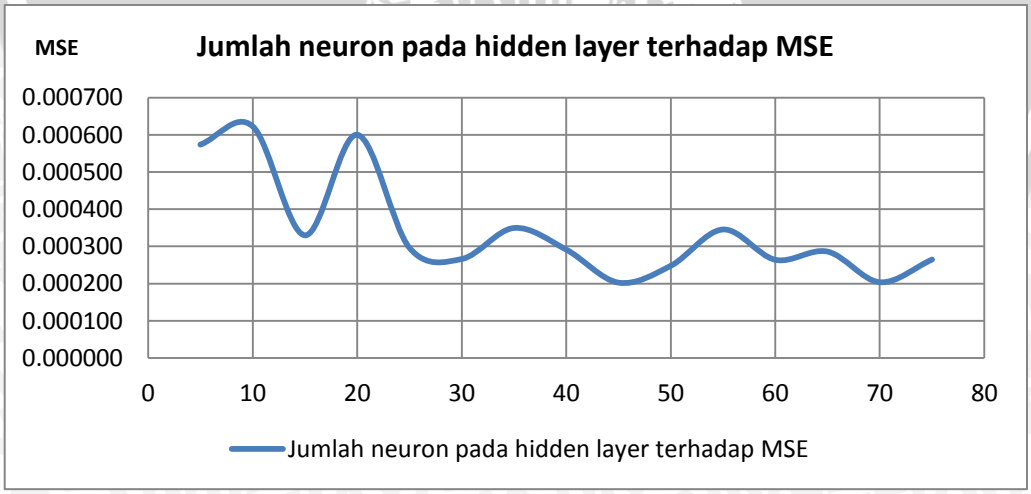
5.1.1 Pengaruh Jumlah Neuron pada Hidden Layer

Percobaan pertama dilakukan dengan mencari jumlah *hidden neuron* optimal. Penentuan jumlah *hidden neuron* pada sistem dilakukan dengan menggunakan proses *trial and error*. Percobaan ini dilakukan dengan cara membandingkan nilai MSE yang diperoleh pada tiap jumlah *hidden neuron*. Jumlah *hidden neuron* yang akan diujikan, yaitu 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75 dengan menggunakan nilai parameter-parameter yang disetting lainnya, yaitu *learning rate* 0.1; *error target* 0.0001; dan maksimum *epoch* sebanyak 5000. Untuk parameter-parameter tetapnya, yaitu faktor penaik 1.2, faktor penurun 0.5, penyesuaian maksimum 50, penyesuaian minimum 0.000001. Semakin kecil nilai MSE, maka konfigurasi jaringan akan semakin baik, sehingga pada percobaan ini akan dipilih nilai MSE terkecil yang akan menghasilkan jumlah *hidden neuron* yang terbaik. Data hasil percobaan untuk mengetahui pengaruh jumlah neuron pada *hidden layer* dapat dilihat pada Tabel 5.1.

Tabel 5. 1 Hasil percobaan pengaruh jumlah neuron pada *hidden layer*

Percobaan ke	Jumlah Neuron pada *Hidden Layer*	MSE
1	5	0.00057350000707110
2	10	0.00062285462847957
3	15	0.00032960819489938
4	20	0.00060012762071136
5	25	0.00029597471680751
6	30	0.00026582556584802
7	35	0.00034929376795830
8	40	0.00029116378757879
9	45	0.00020231351364262
10	50	0.00024752898427662
11	55	0.00034523448107718
12	60	0.00026374290279459
13	65	0.00028573223965807
14	70	0.00020339389792537
15	75	0.00026435232694732

Berdasarkan Tabel 5.1 dapat dilihat bahwa nilai MSE terkecil terletak pada percobaan ke-9, yaitu sebesar 0.00020231351364262 dengan jumlah neuron pada *hidden layer* sebanyak 45 unit. Grafik pengaruh penambahan jumlah neuron pada *hidden layer* terhadap perubahan MSE ditunjukkan pada Gambar 5.1.



Gambar 5. 1 Grafik pengaruh penambahan neuron pada *hidden layer* terhadap perubahan MSE

5.1.2 Pengaruh *Learning Rate*

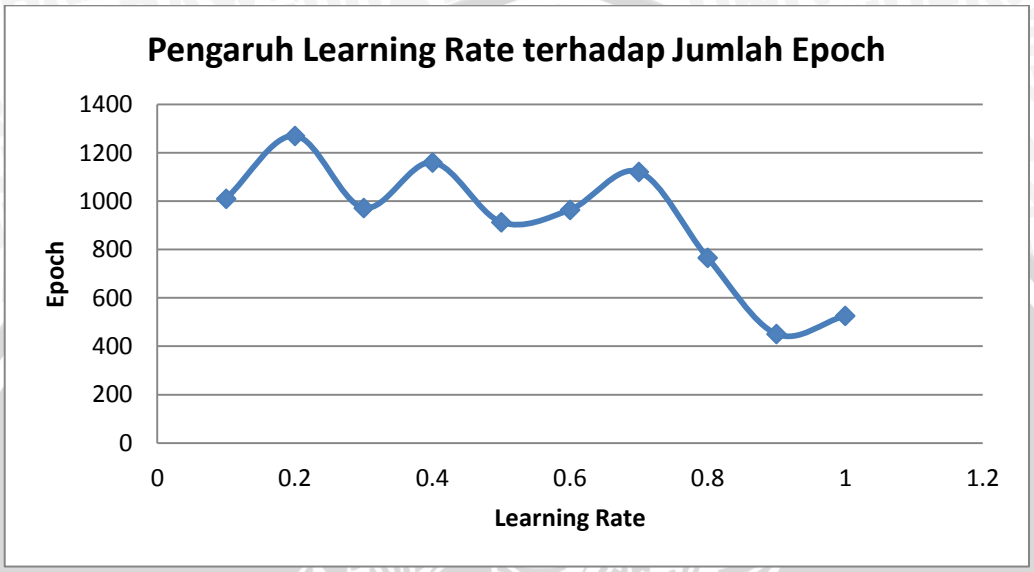
Percobaan ini dilakukan dengan cara membandingkan jumlah *epoch* yang dihasilkan hingga diperoleh nilai MSE < *error target* dengan beberapa nilai *learning rate*, yaitu antara 0 sampai 1. Percobaan dilakukan sebanyak 10 kali dimulai dengan nilai *learning rate* terkecil hingga *learning rate* terbesar, yaitu 0.1 – 1. Jumlah *hidden neuron* yang akan digunakan pada percobaan ini sebanyak 45 unit, karena pada percobaan sebelumnya diketahui konfigurasi jaringan ini lebih optimal.

Nilai parameter-parameter yang disetting lainnya yang digunakan, yaitu *learning rate* 0.1; *error target* 0.001; dan maksimum *epoch* sebanyak 5000. Untuk parameter-parameter tetapnya, yaitu faktor penaik 1.2, faktor penurun 0.5, penyesuaian maksimum 50, penyesuaian minimum 0.000001. Nilai *error target* yang digunakan lebih kecil daripada nilai *error target* pada percobaan sebelumnya. Apabila digunakan nilai *error target* 0.0001, maka kemungkinan besar jaringan akan selalu berhenti pada *epoch* ke 5000, sehingga tidak bisa diketahui seberapa cepat jaringan mencapai bobot yang konvergen berdasarkan nilai *learning rate* yang diuji cobakan. Pada percobaan ini akan dipilih nilai *learning rate* dengan jumlah *epoch* terkecil, karena dengan nilai *learning rate* pada jumlah *epoch* terkecil dapat mempercepat proses pelatihan jaringan. Data hasil percobaan untuk mengetahui pengaruh *learning rate* dapat dilihat pada Tabel 5.2.

Tabel 5. 2 Hasil percobaan pengaruh *learning rate*

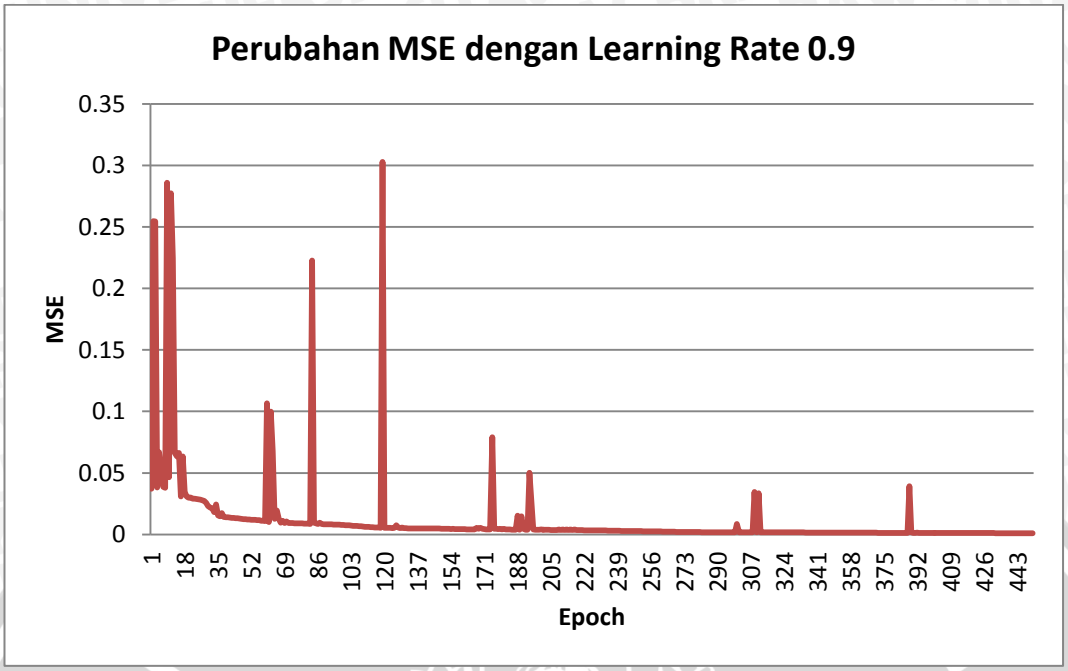
Percobaan ke	*Learning Rate*	Jumlah Epoch
1	0.1	1010
2	0.2	1270
3	0.3	972
4	0.4	1160
5	0.5	913
6	0.6	964
7	0.7	1121
8	0.8	766
9	0.9	451
10	1	526

Grafik pengaruh penambahan *learning rate* terhadap jumlah *epoch* ditunjukkan pada Gambar 5.2. Berdasarkan Gambar 5.2 dapat dilihat bahwa penambahan jumlah *epoch* yang dicapai saat pelatihan tidak berbanding lurus dengan peningkatan nilai *learning rate*.



Gambar 5. 2 Grafik pengaruh penambahan *learning rate* terhadap jumlah *epoch*

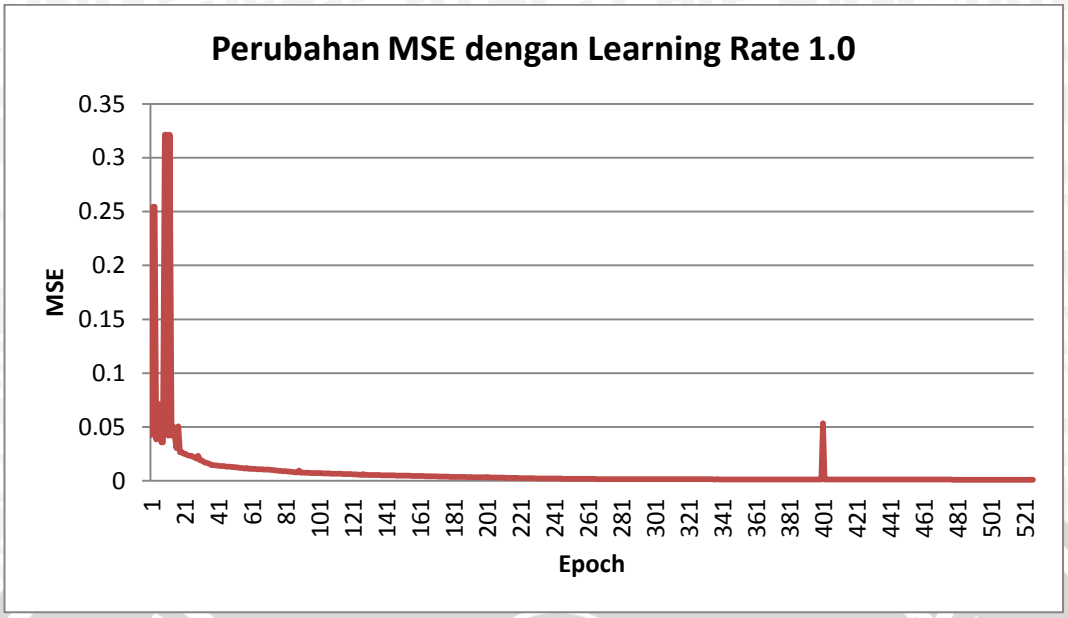
Berdasarkan Tabel 5.2 dapat dilihat bahwa jumlah *epoch* terkecil terletak pada percobaan ke-9, yaitu sebesar 451 *epoch* dengan nilai *learning rate* sebesar 0.9. Grafik hasil pelatihan dengan *learning rate* 0.9 dengan jumlah *epoch* yang dicapai 451 dapat dilihat pada Gambar 5.3. Hal ini menunjukkan bahwa dengan menggunakan *learning rate* dengan *epoch* terkecil saat pelatihan tidak menjamin nilai MSE yang dihasilkan stabil, karena ternyata nilai *error* (MSE) yang dihasilkan naik turun tidak terkendali.



Gambar 5. 3 Grafik perubahan MSE dengan *learning rate* 0.9

Karena dengan *learning rate* 0.9 tidak memungkinkan untuk dijadikan sebagai nilai parameter terbaik dalam konfigurasi jaringan, maka dipilih satu kandidat nilai *learning rate* terbaik dengan jumlah *epoch* terkecil kedua setelah 451, yaitu terdapat pada percobaan ke-10 dengan 526 *epoch* pada saat *learning rate* sebesar 1.0.

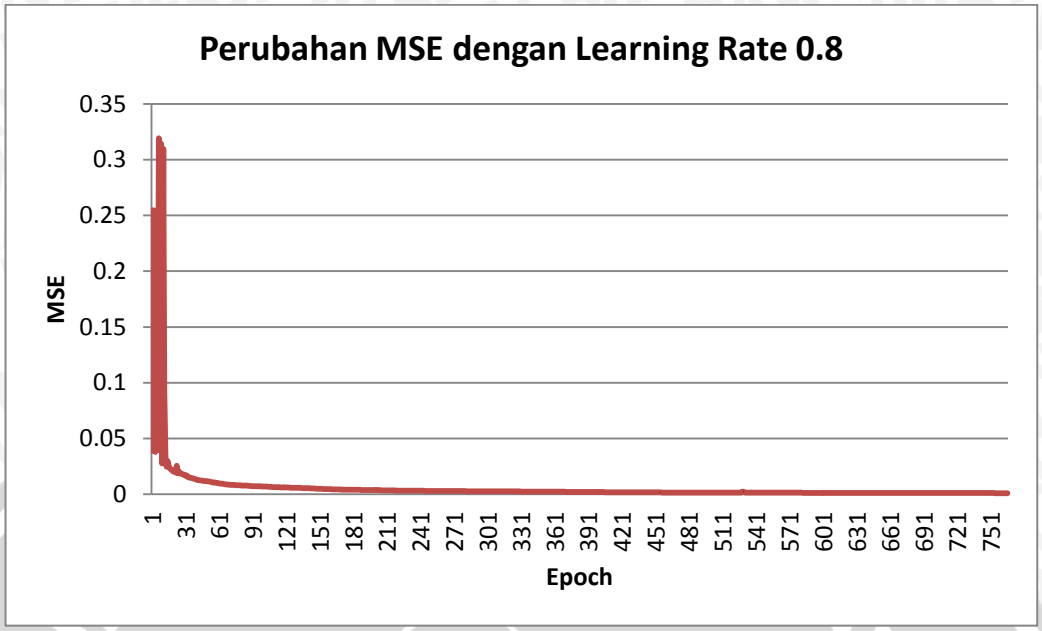
Grafik hasil pelatihan dengan *learning rate* 1.0 dengan jumlah *epoch* yang dicapai 526 dapat dilihat pada Gambar 5.4. Hal ini menunjukkan bahwa dengan menggunakan *learning rate* 1.0, pada awal *epoch* pelatihan terjadi penurunan *error* (MSE) terlalu cepat dan kemudian di saat *epoch* 401 terlihat adanya peningkatan nilai MSE yang cukup drastis kemudian turun secara drastis lagi saat *epoch* 402, sehingga perubahan MSE menjadi kurang stabil.



Gambar 5. 4 Grafik perubahan MSE dengan *learning rate* 1.0

Karena dengan *learning rate* 1.0 tidak memungkinkan untuk dijadikan sebagai nilai parameter terbaik dalam konfigurasi jaringan, maka dipilih satu kandidat nilai *learning rate* terbaik dengan jumlah *epoch* terkecil ketiga setelah 526, yaitu terdapat pada percobaan ke-8 dengan 766 *epoch* pada saat *learning rate* sebesar 0.8.

Grafik hasil pelatihan dengan *learning rate* 0.8 dengan jumlah *epoch* yang dicapai 766 dapat dilihat pada Gambar 5.5. Hal ini menunjukkan bahwa dengan menggunakan *learning rate* 0.8, tidak adanya penurunan *error* (MSE) yang terlalu lambat ataupun terlalu cepat. Pada awal iterasi memang terjadi penurunan MSE yang terlalu cepat, akan tetapi hal itu tidak berlangsung lama. Setelah itu penurunan MSE berlangsung tidak terlalu cepat maupun tidak terlalu lambat saat *epoch* ke 12 hingga seterusnya, sehingga perubahan MSE masih stabil. Oleh karena itu dipilihlah nilai *learning rate* 0.8 sebagai nilai parameter untuk konfigurasi jaringan optimal.



Gambar 5. 5 Grafik perubahan MSE dengan *learning rate* 0.8

5.2 Hasil Pengujian

Hasil dari pelatihan berupa bobot dan bias yang telah tersimpan pada dataset yang akan digunakan untuk menguji kemampuan sistem dalam memberikan prediksi klasifikasi status tahapan keluarga sejahtera. Pada pengujian ini yang dilakukan adalah membandingkan keluaran sistem dengan data aktual, sehingga dapat diketahui nilai akurasi keberhasilan sistem. Pada hasil percobaan yang telah dilakukan sebelumnya telah menghasilkan konfigurasi jaringan yang optimal, yaitu dengan kombinasi jumlah *hidden neuron* sebanyak 45 unit dan *learning rate* sebesar 0.8. Konfigurasi jaringan ditunjukkan pada Tabel 5.3.

Tabel 5. 3 Konfigurasi jaringan syaraf tiruan yang digunakan

Parameter	Nilai
Jumlah neuron pada *input layer*	21
Jumlah neuron pada *hidden layer*	45
Jumlah neuron pada *output layer*	1
Error target	0.0001
Learning rate	0.8
Faktor penaik	1.2
Faktor penurun	0.5
Penyesuaian maksimum	50
Penyesuaian minimum	$0.000001 = 10^{-6}$
Maksimum *epoch*	5000

5.2.1 Pengaruh Jumlah Data Latih Seimbang

Pengujian pada pengaruh jumlah data latih seimbang dilakukan dengan data latih yang berbeda-beda. Dalam pengujian ini data latih dalam keadaan kelas data seimbang *(Balanced Class)* artinya tidak ada kelas yang lebih mendominasi. Pembagian data yang digunakan untuk pelatihan dan pengujian data sebanyak 200, dibagi menjadi 2, yaitu 70% (140 data) untuk data latih dan 30% (60 data) untuk data uji. Variasi data latihnya, yaitu dikurangi setiap 10% data dari 70% data latih, sehingga secara berturut-turut data latih yang digunakan adalah sebanyak 20 data (10%), 40 data (20%), 60 data (30%), 80 data (40%), 100 data (50%), 120 data (60%), dan 140 data (70%). Data uji yang digunakan adalah sebanyak 30% (60 data) yang belum pernah dilatihkan sebelumnya. Pelatihan dan pengujian ini dilakukan sebanyak 5 kali percobaan pada masing-masing proporsi data latih. Percobaan ke-1 sampai ke-5 dilakukan dengan cara menguji data yang sama dengan data uji 30% (60 data) dan menggunakan konfigurasi jaringan optimal yang didapatkan dari pelatihan yang sudah dilakukan pada percobaan sebelumnya yang hasilnya ditunjukkan pada Tabel 5.3. Hasil pengujian pengaruh jumlah data latih seimbang terhadap akurasi sitem ditunjukkan pada Tabel 5.4 sampai Tabel 5.10.

Tabel 5. 4 Hasil percobaan pengaruh data latih seimbang 10% terhadap akurasi

Kategori	Data Latih 10%	Data Uji 30%	Percobaan ke									
			1		2		3		4		5	
	(20 data)	(60 data)	✔	✘	✔	✘	✔	✘	✔	✘	✔	✘
KPS	4	12	12	0	11	1	10	2	11	1	11	1
KS I	4	12	4	8	6	5	2	10	2	10	3	9
KS II	4	12	10	2	10	2	10	2	11	1	10	2
KS III	4	12	11	1	11	1	11	1	11	1	11	1
KS III+	4	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			49		51		45		47		47	
Jumlah Prediksi Salah			11		9		15		13		13	
Akurasi			81.67%		85.00%		75.00%		78.33%		78.33%	
Rata-rata Akurasi			79.67%									
Akurasi Tertinggi			85.00%									

Tabel 5. 5 Hasil percobaan pengaruh data latih seimbang 20% terhadap akurasi

Kategori	Data Latih 20%	Data Uji 30%	Percobaan ke									
			1		2		3		4		5	
	(40 data)	(60 data)	✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	8	12	11	1	11	1	10	2	10	2	11	1
KS I	8	12	9	3	5	7	9	3	9	3	9	3
KS II	8	12	7	5	8	4	7	5	6	6	8	4
KS III	8	12	11	1	11	1	12	0	11	1	12	0
KS III+	8	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			50		47		50		48		52	
Jumlah Prediksi Salah			10		13		10		12		8	
Akurasi			83.33%		78.33%		83.33%		80.00%		86.67%	
Rata-rata Akurasi			82.33%									
Akurasi Tertinggi			86.67%									

Tabel 5. 6 Hasil percobaan pengaruh data latih seimbang 30% terhadap akurasi

Kategori	Data Latih 30%	Data Uji 30%	Percobaan ke									
			1		2		3		4		5	
	(60 data)	(60 data)	✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	12	12	12	0	12	0	12	0	12	0	12	0
KS I	12	12	7	5	10	2	8	4	9	3	9	3
KS II	12	12	8	4	6	6	8	4	7	5	7	5
KS III	12	12	11	1	11	1	11	1	12	0	12	0
KS III+	12	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			50		51		51		52		52	
Jumlah Prediksi Salah			10		9		9		8		8	
Akurasi			83.33%		85.00%		85.00%		86.67%		86.67%	
Rata-rata Akurasi			85.33%									
Akurasi Tertinggi			86.67%									

Tabel 5. 7 Hasil percobaan pengaruh data latih seimbang 40% terhadap akurasi

Kategori	Data Latih 40% (80 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	16	12	12	0	12	0	12	0	12	0	12	0
KS I	16	12	10	2	9	3	10	2	8	4	8	4
KS II	16	12	5	7	8	4	8	4	7	5	8	4
KS III	16	12	12	0	12	0	12	0	12	0	12	0
KS III+	16	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			51		53		54		51		52	
Jumlah Prediksi Salah			9		7		6		9		8	
Akurasi			85.00%		88.33%		90.00%		85.00%		86.67%	
Rata-rata Akurasi			87.00%									
Akurasi Tertinggi			90.00%									

Tabel 5. 8 Hasil percobaan pengaruh data latih seimbang 50% terhadap akurasi

Kategori	Data Latih 50% (100 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	20	12	12	0	12	0	12	0	12	0	12	0
KS I	20	12	8	4	9	3	9	3	8	4	9	3
KS II	20	12	7	5	9	3	8	4	8	4	8	4
KS III	20	12	12	0	12	0	12	0	12	0	12	0
KS III+	20	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			51		54		53		52		53	
Jumlah Prediksi Salah			9		6		7		8		7	
Akurasi			85.00%		90.00%		88.33%		86.67%		88.33%	
Rata-rata Akurasi			87.67%									
Akurasi Tertinggi			90.00%									

Tabel 5. 9 Hasil percobaan pengaruh data latih seimbang 60% terhadap akurasi

Kategori	Data Latih 60% (120 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	24	12	12	0	12	0	12	0	12	0	12	0
KS I	24	12	9	3	9	3	10	2	9	3	9	3
KS II	24	12	8	4	8	4	9	3	8	4	8	4
KS III	24	12	12	0	12	0	12	0	12	0	12	0
KS III+	24	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			53		53		55		53		53	
Jumlah Prediksi Salah			7		7		5		7		7	
Akurasi			88.33%		88.33%		91.67%		88.33%		88.33%	
Rata-rata Akurasi			89.00%									
Akurasi Tertinggi			91.67%									

Tabel 5. 10 Hasil percobaan pengaruh data latih seimbang 70% terhadap akurasi

Kategori	Data Latih 70% (140 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	28	12	12	0	12	0	12	0	12	0	12	0
KS I	28	12	9	3	10	2	9	3	11	1	10	2
KS II	28	12	8	4	10	2	8	4	9	3	7	5
KS III	28	12	12	0	12	0	12	0	12	0	12	0
KS III+	28	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			53		56		53		56		53	
Jumlah Prediksi Salah			7		4		7		4		7	
Akurasi			88.33%		93.33%		88.33%		93.33%		88.33%	
Rata-rata Akurasi			90.33%									
Akurasi Tertinggi			93.33%									

Keterangan :

✓ : Jumlah prediksi benar

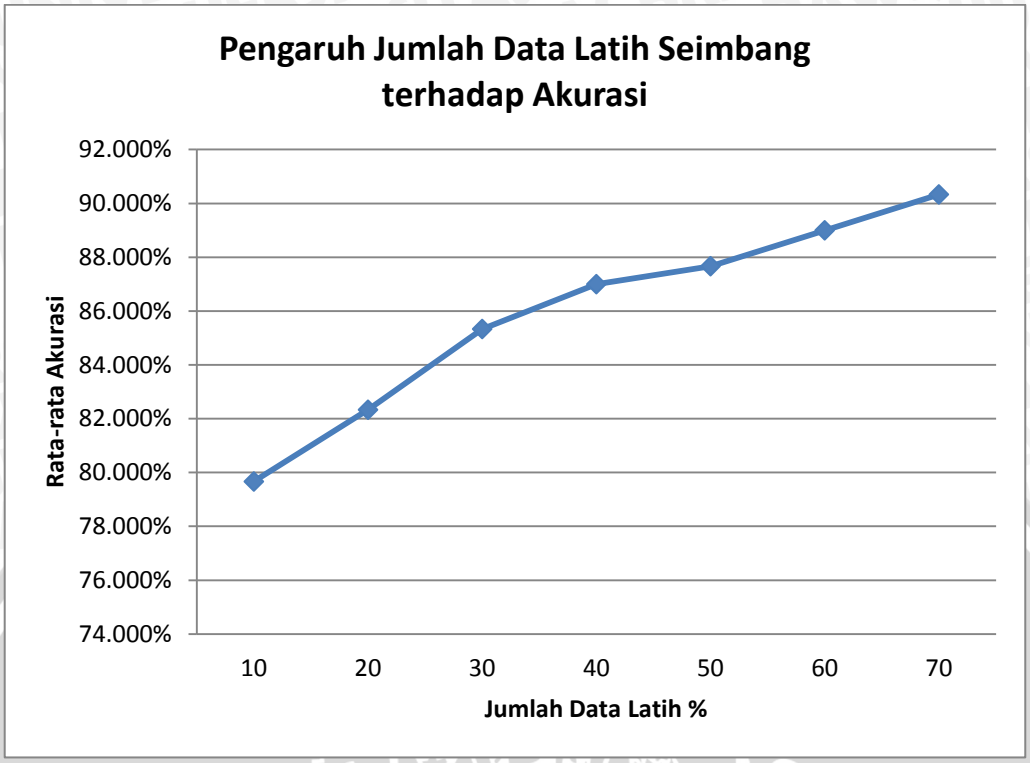
✗ : Jumlah prediksi salah

Berdasarkan hasil percobaan pengaruh jumlah data latih seimbang terhadap akurasi yang ditunjukkan pada Tabel 5.4 sampai Tabel 5.10, maka dapat diketahui bahwa akurasi tertinggi dari semua jenis jumlah data latih yang diujikan adalah pada data latih 70%, yaitu 93.33% begitu juga dengan rata-rata akurasi yang tertinggi juga terdapat pada saat data latih 70%, yaitu 90.33%. Pada Tabel 5.11 disajikan akurasi rata-rata dan akurasi tertinggi dari 5 kali percobaan pada setiap jumlah data latih.

Tabel 5. 11 Hasil percobaan pengaruh data latih seimbang terhadap akurasi

Jumlah Data Latih (%)	Jumlah Data Latih	Jumlah Data Uji (30%)	Rata-rata Akurasi dari Percobaan 1 sampai 5	Akurasi Tertinggi dari Percobaan 1 sampai 5
10	20	60	79.67%	85.00%
20	40	60	82.33%	86.67%
30	60	60	85.33%	86.67%
40	80	60	87.00%	90.00%
50	100	60	87.67%	90.00%
60	120	60	89.00%	91.67%
70	140	60	90.33%	93.33%

Hal ini menunjukkan bahwa untuk penggunaan sistem klasifikasi status tahapan keluarga sejahtera dengan JST *Resilient backpropagation*, akurasi terbaik akan tercapai apabila menggunakan jumlah data latih sebesar 70% dalam kondisi kelas seimbang (140 data) dan data uji 30% (60 data). Grafik hasil percobaan pada pengaruh jumlah data latih seimbang terhadap rata-rata akurasi sistem berdasarkan 5 kali percobaan ditunjukkan pada Gambar 5.6.

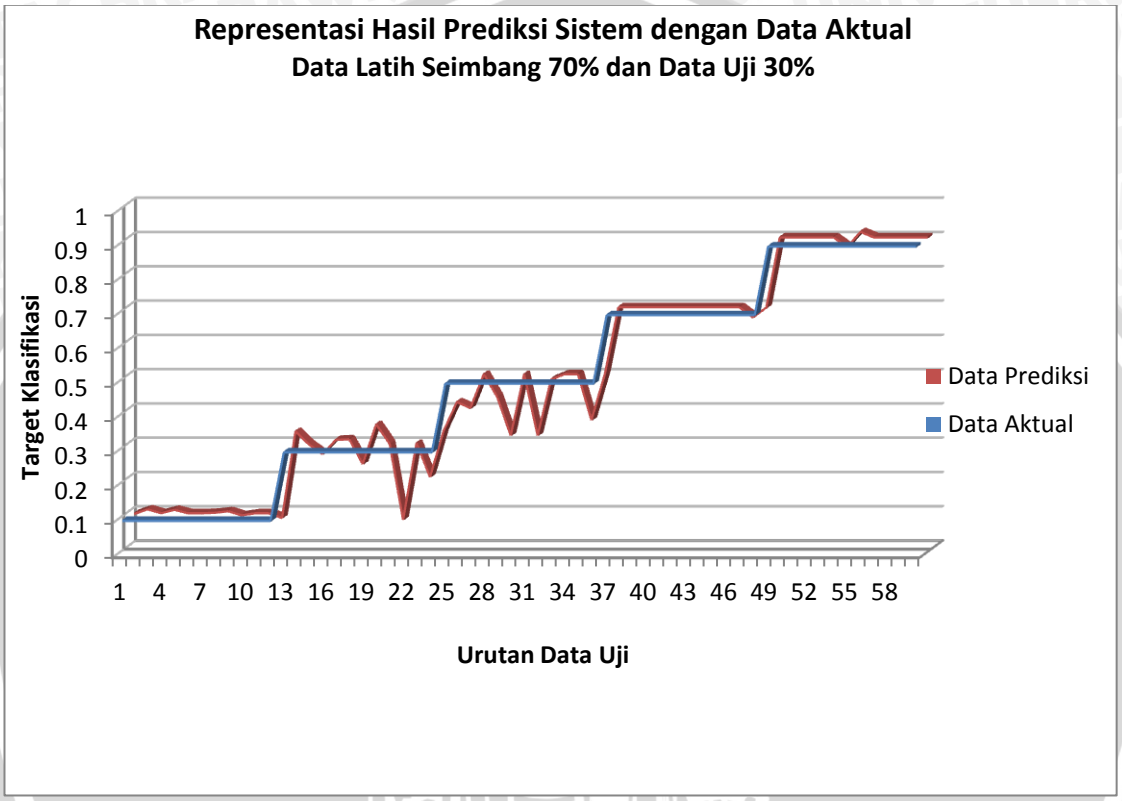


Gambar 5. 6 Grafik hasil percobaan pengaruh jumlah data latih seimbang terhadap rata-rata akurasi sistem

Diketahui bahwa akurasi tertinggi dari semua jenis jumlah data latih seimbang yang diujikan adalah pada data latih 70%, yaitu 93.33%. Berdasarkan hal ini maka dapat dibuat representasi hasil prediksi sistem terbaik dengan data aktual yang ada. Grafik representasi hasil prediksi sistem dengan data aktual dapat dilihat pada Gambar 5.7. Berdasarkan grafik pada Gambar 5.7 ini dapat diketahui selisih nilai antara data prediksi sistem dengan data aktual yang ada.

Selisih nilai di antara keduanya yang paling besar adalah 0.221942389 pada data uji ke-21, sedangkan selisih nilai di antara keduanya yang paling kecil adalah 0.000606356 pada data uji ke-20. Selisih nilai yang paling besar ini menunjukkan sistem salah memprediksi data. Kesalahan prediksi yang paling besar nilainya ini terjadi pada data uji ke-21, padahal data aktual menunjukkan kategori kelas "KS I" dengan nilai 0.3, sedangkan hasil prediksi menunjukkan kategori kelas "Prasejahtera" dengan nilai 0.078057611. Selain itu, kesalahan prediksi juga terdapat pada data uji ke-29, 31, dan 35 dengan masing-masing nilai data aktual adalah 0.5 tergolong ke dalam kategori kelas KS II dan nilai prediksi

yang dihasilkan masing-masing secara berturut-turut adalah 0.323632797 (KS I), 0.323632797 (KS I), dan 0.368769789 (KS I). dengan masing-masing berturut-turut selisih nilai antara data prediksi dengan data actual adalah 0.176367203; 0.176367203; dan 0.131230211.



Gambar 5. 7 Grafik representasi hasil prediksi sistem dengan data aktual saat jumlah data latih seimbang

5.2.2 Pengaruh Jumlah Data Latih Tidak Seimbang

Pengujian pada pengaruh jumlah data latih tidak seimbang dilakukan dengan data latih yang berbeda-beda. Dalam pengujian ini data latih dalam keadaan kelas data tidak seimbang *(Unbalanced Class)* artinya jumlah data pada masing-masing kelas dibuat tidak seimbang atau tidak sama. Pembagian data yang digunakan untuk pelatihan dan pengujian sama seperti pada pengujian jumlah data latih seimbang, yaitu dengan data sebanyak 200, dibagi menjadi 2, yaitu 70% (140 data) untuk data latih dan 30% (60 data) untuk data uji. Variasi data latihnya, yaitu dikurangi setiap 10% data dari 70% data latih, sehingga secara berturut-turut

data latih yang digunakan adalah sebanyak 20 data (10%), 40 data (20%), 60 data (30%), 80 data (40%), 100 data (50%), 120 data (60%), dan 140 data (70%). Data uji yang digunakan adalah sebanyak 30% (60 data) yang belum pernah dilatihkan sebelumnya. Pelatihan dan pengujian ini dilakukan sebanyak 5 kali percobaan pada masing-masing proporsi data latih. Percobaan ke-1 sampai ke-5 dilakukan dengan cara menguji data yang sama dengan data uji 30% (60 data) dan menggunakan konfigurasi jaringan optimal yang didapatkan dari pelatihan yang sudah dilakukan pada percobaan sebelumnya yang hasilnya ditunjukkan pada Tabel 5.3. Hasil pengujian pengaruh jumlah data latih tidak seimbang terhadap akurasi sitem ditunjukkan pada Tabel 5.12 sampai Tabel 5.18.

Tabel 5. 12 Hasil percobaan pengaruh data latih tidak seimbang 10% terhadap akurasi

Kategori	Data Latih 10%	Data Uji 30%	Percobaan ke									
			1		2		3		4		5	
	(20 data)	(60 data)	✅	❌	✅	❌	✅	❌	✅	❌	✅	❌
KPS	2	12	8	4	8	4	9	3	9	3	9	3
KS I	3	12	6	6	8	4	7	5	9	3	8	4
KS II	6	12	10	2	10	2	10	2	10	2	10	2
KS III	5	12	11	1	11	1	11	1	11	1	11	1
KS III+	4	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			47		49		49		51		50	
Jumlah Prediksi Salah			13		11		11		9		10	
Akurasi			78.33%		81.67%		81.67%		85.00%		83.33%	
Rata-rata Akurasi			82.00%									
Akurasi Tertinggi			85.00%									

Tabel 5. 13 Hasil percobaan pengaruh data latih tidak seimbang 20% terhadap akurasi

Kategori	Data Latih 20%	Data Uji 30%	Percobaan ke									
			1		2		3		4		5	
	(40 data)	(60 data)	✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	4	12	11	1	11	1	10	2	10	2	11	1
KS I	6	12	8	4	8	4	9	3	9	3	4	8
KS II	12	12	10	2	11	1	12	0	7	5	11	1
KS III	10	12	11	1	12	0	12	0	12	0	12	0
KS III+	8	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			52		54		55		50		50	
Jumlah Prediksi Salah			8		6		5		10		10	
Akurasi			86.67%		90.00%		91.67%		83.33%		83.33%	
Rata-rata Akurasi			87.00%									
Akurasi Tertinggi			91.67%									

Tabel 5. 14 Hasil percobaan pengaruh data latih tidak seimbang 30% terhadap akurasi

Kategori	Data Latih 30%	Data Uji 30%	Percobaan ke									
			1		2		3		4		5	
	(60 data)	(60 data)	✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	8	12	11	1	10	2	11	1	12	0	11	1
KS I	12	12	9	3	8	4	10	2	10	2	9	3
KS II	20	12	7	5	8	4	8	4	8	4	9	3
KS III	12	12	12	0	12	0	12	0	12	0	12	0
KS III+	8	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			51		50		53		54		53	
Jumlah Prediksi Salah			9		10		7		6		7	
Akurasi			85.00%		83.33%		88.33%		90.00%		88.33%	
Rata-rata Akurasi			87.50%									
Akurasi Tertinggi			90.00%									

Tabel 5. 15 Hasil percobaan pengaruh data latih tidak seimbang 40% terhadap akurasi

Kategori	Data Latih 40% (80 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	16	12	12	0	12	0	12	0	12	0	12	0
KS I	15	12	8	4	8	4	8	4	8	4	9	3
KS II	20	12	8	4	8	4	7	5	7	5	6	6
KS III	25	12	12	0	12	0	12	0	12	0	12	0
KS III+	4	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			52		52		51		51		51	
Jumlah Prediksi Salah			8		8		9		9		9	
Akurasi			86.67%		86.67%		85.00%		85.00%		85.00%	
Rata-rata Akurasi			85.67%									
Akurasi Tertinggi			86.67%									

Tabel 5. 16 Hasil percobaan pengaruh data latih tidak seimbang 50% terhadap akurasi

Kategori	Data Latih 50% (100 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	17	12	12	0	12	0	12	0	12	0	12	0
KS I	20	12	8	4	9	3	9	3	8	4	8	4
KS II	25	12	8	4	9	3	8	4	8	4	8	4
KS III	30	12	12	0	12	0	12	0	12	0	12	0
KS III+	8	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			52		54		53		52		52	
Jumlah Prediksi Salah			8		6		7		8		8	
Akurasi			86.67%		90.00%		88.33%		86.67%		86.67%	
Rata-rata Akurasi			87.67%									
Akurasi Tertinggi			90.00%									

Tabel 5. 17 Hasil percobaan pengaruh data latih tidak seimbang 60% terhadap akurasi

Kategori	Data Latih 60% (120 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	20	12	12	0	12	0	12	0	12	0	12	0
KS I	25	12	8	4	9	3	9	3	9	3	9	3
KS II	30	12	10	2	8	4	10	2	8	4	9	3
KS III	35	12	12	0	12	0	12	0	12	0	12	0
KS III+	10	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			54		53		55		53		54	
Jumlah Prediksi Salah			6		7		5		7		6	
Akurasi			90.00%		88.33%		91.67%		88.33%		90.00%	
Rata-rata Akurasi			89.67%									
Akurasi Tertinggi			91.67%									

Tabel 5. 18 Hasil percobaan pengaruh data latih tidak seimbang 70% terhadap akurasi

Kategori	Data Latih 70% (140 data)	Data Uji 30% (60 data)	Percobaan ke 1		2		3		4		5	
			✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
KPS	20	12	12	0	12	0	12	0	12	0	12	0
KS I	27	12	9	3	10	2	9	3	10	2	9	3
KS II	37	12	10	2	9	3	8	4	8	4	8	4
KS III	40	12	12	0	12	0	12	0	12	0	12	0
KS III+	16	12	12	0	12	0	12	0	12	0	12	0
Jumlah Prediksi Benar			55		55		53		54		53	
Jumlah Prediksi Salah			5		5		7		6		7	
Akurasi			91.67%		91.67%		88.33%		90.00%		88.33%	
Rata-rata Akurasi			90.00%									
Akurasi Tertinggi			91.67%									

Keterangan :

✓ : Jumlah prediksi benar

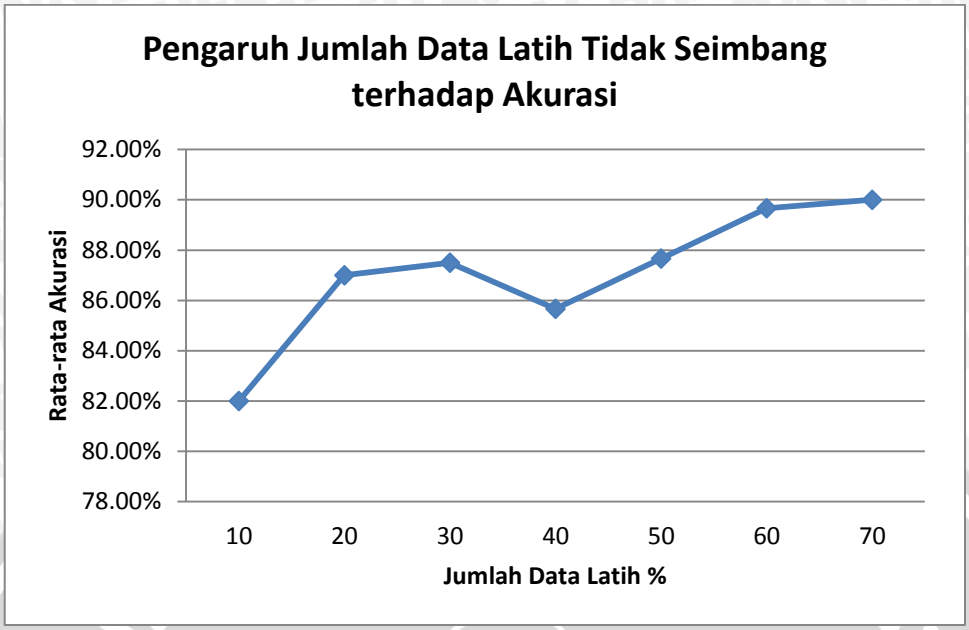
✗ : Jumlah prediksi salah

Berdasarkan hasil percobaan pengaruh jumlah data latih tidak seimbang terhadap akurasi yang ditunjukkan pada Tabel 5.4 sampai Tabel 5.10, maka dapat diketahui bahwa akurasi tertinggi dari semua jenis jumlah data latih yang diujikan adalah pada data latih 70%, yaitu 91.67% begitu juga dengan rata-rata akurasi yang tertinggi juga terdapat pada saat data latih 70%, yaitu 90.00%. Pada Tabel 5.19 disajikan akurasi rata-rata dan akurasi tertinggi dari 5 kali percobaan pada setiap jumlah data latih.

Tabel 5. 19 Hasil percobaan pengaruh data latih tidak seimbang terhadap akurasi

Jumlah Data Latih (%)	Jumlah Data Latih	Jumlah Data Uji (30%)	Rata-rata Akurasi dari Percobaan 1 sampai 5	Akurasi Tertinggi dari Percobaan 1 sampai 5
10	20	60	82.00%	85.00%
20	40	60	87.00%	91.67%
30	60	60	87.50%	90.00%
40	80	60	85.67%	86.67%
50	100	60	87.67%	90.00%
60	120	60	89.67%	91.67%
70	140	60	90.00%	91.67%

Hal ini menunjukkan bahwa untuk penggunaan sistem klasifikasi status tahapan keluarga sejahtera dengan JST *Resilient backpropagation*, maka akurasi terbaik akan tercapai apabila menggunakan jumlah data latih tidak seimbang sebesar 70% (140 data) dan data uji 30% (60 data). Grafik hasil percobaan pada pengaruh jumlah data latih tidak seimbang terhadap rata-rata akurasi sistem berdasarkan 5 kali percobaan ditunjukkan pada Gambar 5.8.

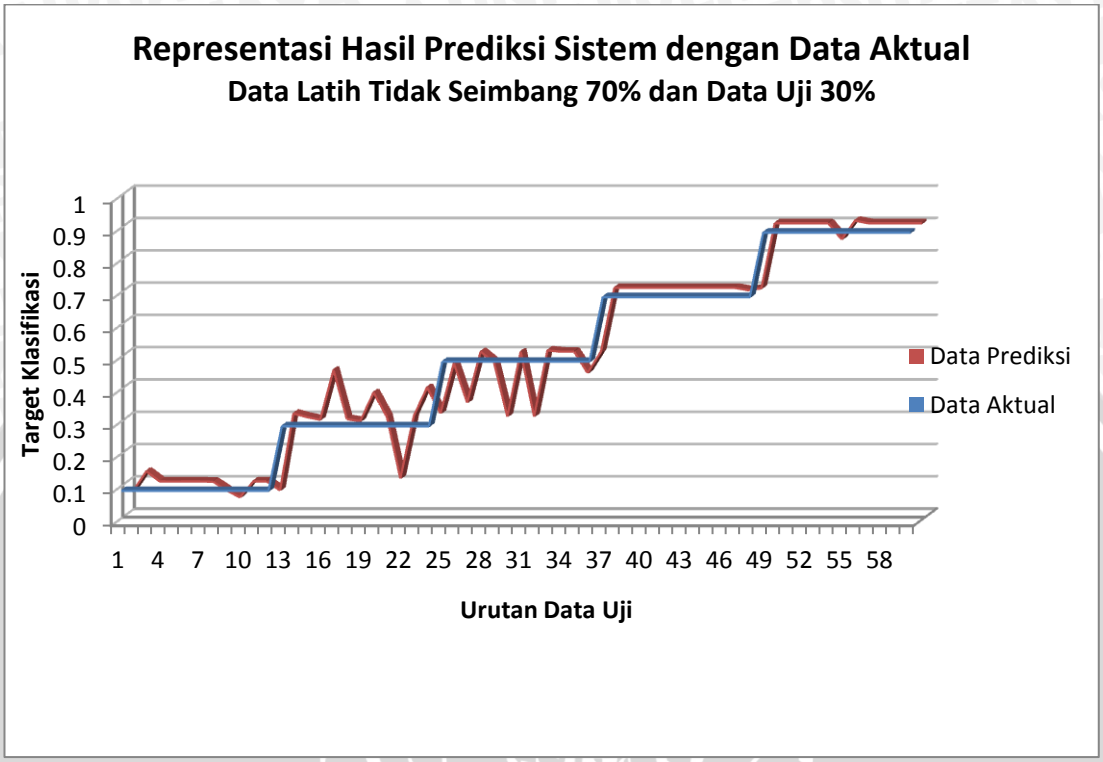


Gambar 5. 8 Grafik hasil percobaan pengaruh jumlah data latih tidak seimbang terhadap rata-rata akurasi sistem

Diketahui bahwa akurasi tertinggi dari semua jenis jumlah data latih tidak seimbang yang diujikan adalah pada data latih 70%, yaitu 91.67%. Berdasarkan hal ini maka dapat dibuat representasi hasil prediksi sistem terbaik dengan data aktual yang ada. Grafik representasi hasil prediksi sistem dengan data aktual dapat dilihat pada Gambar 5.9. Berdasarkan grafik pada Gambar 5.9 ini dapat diketahui selisih nilai antara data prediksi sistem dengan data aktual yang ada.

Selisih nilai di antara keduanya yang paling besar adalah 0.196618053 pada data uji ke-31, sedangkan selisih nilai di antara keduanya yang paling kecil adalah 0.000845863 pada data uji ke-48. Selisih nilai yang paling besar ini menunjukkan sistem salah memprediksi data. Kesalahan prediksi yang paling besar nilainya ini terjadi pada data uji ke-31, padahal data aktual menunjukkan kategori kelas "KS I" dengan nilai 0.3, sedangkan hasil prediksi menunjukkan kategori kelas "Prasejahtera" dengan nilai 0.054079496. Selain itu, kesalahan prediksi juga terdapat pada data uji ke-16, 21, 26, dan 29 dengan masing-masing nilai data aktual adalah 0.3 (KS I), 0.3 (KS I), 0.5 (KS II), 0.5 (KS II) dan nilai prediksi yang dihasilkan masing-masing secara berturut-turut adalah 0.448190191 (KS II), 0.111882478 (KPS), 0.346639342 (KS I), 0.303381947 (KS I) dengan

masing-masing berturut-turut selisih nilai antara data prediksi dengan data aktualnya adalah 0.148190191; 0.188117522; 0.153360658; dan 0.196618053.



Gambar 5. 9 Grafik representasi hasil prediksi sistem dengan data aktual saat jumlah data latih tidak seimbang

5.3 Analisa Hasil

Untuk memperoleh sistem yang memiliki tingkat akurasi tinggi ditentukan pula oleh pemilihan konfigurasi jaringan dan parameter pelatihan jaringan yang tepat. Oleh karena itu, dilakukan analisis terhadap konfigurasi jaringan berdasarkan parameternya. Jaringan syaraf tiruan *resilient backpropagation* dengan konfigurasi jaringan 21 unit neuron pada *input layer*, 45 unit neuron pada *hidden layer*, 1 unit neuron pada *output layer*, dan *learning rate* 0.8 mampu memberikan tingkat keberhasilan yang cukup bagus dalam memberikan prediksi klasifikasi status tahapan keluarga sejahtera. Parameter lainnya yang harus disetting, yaitu *error target* 0.0001, dan *max epoch* 5000. Untuk parameter-parameter tetapnya, yaitu faktor penaik 1.2, faktor penurun 0.5, penyesuaian maksimum 50, dan penyesuaian minimum 0.000001.

5.3.1 Analisa Hasil Pengaruh Jumlah Neuron pada *Hidden Layer*

Berdasarkan hasil pelatihan dengan jumlah *neuron hidden* yang berbeda-beda pada Tabel 5.1 dan Gambar 5.1 dapat dilihat bahwa MSE yang dihasilkan pada masing-masing percobaan selalu berubah-ubah. Hal ini disebabkan karena bobot dan bias awal yang digunakan merupakan nilai acak (random). Berdasarkan pada Tabel 5.1 dan Gambar 5.1 ditunjukkan bahwa percobaan pada penambahan jumlah *hidden neuron* tidak berbanding lurus pada peningkatan atau penurunan MSE. Ketika data latih dilatihkan dengan *hidden neuron* 5 sampai 20 nilai MSE masih naik turun, karena kemampuan generalisasi jaringan yang masih belum stabil. Namun, sebenarnya ketika jaringan dilatihkan dengan *hidden neuron* 15, bisa dianggap jaringan sudah mulai stabil, karena memiliki nilai MSE sebesar 0.00032960819489938 yang mendekati nilai *error target Hidden neuron* yang berjumlah 15 unit dihasilkan dari perhitungan pada Persamaan (2-8). Hal ini membuktikan bahwa dengan menggunakan Persamaan (2-8) untuk menentukan *hidden neuron* bisa dianggap mampu menstabilkan jaringan. Namun, akan lebih baik lagi apabila penentuan *hidden neuron* menggunakan metode *trial and error*, karena dengan metode ini bisa diketahui kemampuan generalisasi jaringan berdasarkan nilai MSE.

Berdasarkan pada hasil percobaan pada Tabel 5.1, maka peneliti memilih jumlah *hidden neuron* sebanyak 45 unit, karena memiliki nilai MSE terkecil dari semua *hidden neuron* yang dicobakan. Pada saat jaringan dilatihkan dengan *hidden neuron* 25 sampai 75, nilai MSE naik turun, tetapi naik turunnya nilai MSE tersebut tidak terlalu terpaut jauh. Hal ini disebabkan jumlah *hidden neuron* yang besar membuat jaringan lebih fleksibel dan dapat menghasilkan MSE yang kecil, tetapi jika jumlah *hidden neuron* terlalu besar akan menghasilkan MSE yang tidak stabil, selain itu waktu yang dibutuhkan untuk pelatihan jaringan lebih lama.

5.3.2 Analisa Hasil Pengaruh *Learning Rate*

Hasil pelatihan dengan *learning rate* yang berbeda-beda ditunjukkan pada Tabel 5.2 dan Gambar 5.2. Berdasarkan hasil pelatihan tersebut dapat dilihat bahwa peningkatan nilai *learning rate* tidak berbanding lurus dengan penambahan

jumlah *epoch* yang dicapai. Hal ini disebabkan karena bobot dan bias awal yang digunakan merupakan nilai acak (random). Apabila nilai bobot dan bias awal yang digunakan terlalu besar atau terlalu kecil dapat menyebabkan perubahan bobot menjadi sangat kecil, sehingga menyebabkan *epoch* yang dibutuhkan untuk mencapai *error target* (konvergen) akan semakin banyak. Perubahan bobot pada algoritma RPROP dipengaruhi oleh besarnya nilai *learning rate*, karena nilai *learning rate* ini akan selalu diupdate berdasarkan nilai gradiennya. Apabila perubahan bobot yang dibuat terlalu besar, maka akan terjadi nilai *error* (MSE) yang naik turun.

Berdasarkan hasil percobaan pada Tabel 5.2, maka dipilih nilai *learning rate* sebesar 0.8, karena grafik perubahan nilai MSE-nya yang masih stabil ditunjukkan pada Gambar 5.5. Selain itu, *learning rate* sebesar 0.8 memiliki jumlah *epoch* terkecil ketiga setelah dua kandidat lainnya (*learning rate* 0.9 dan 1.0) tidak memenuhi kriteria dikarenakan grafik perubahan MSE-nya masih kurang stabil.

Grafik konvergensi pelatihan pada saat *learning rate* 0.9 yang ditunjukkan pada Gambar 5.3 menunjukkan bahwa grafik tersebut terlihat adanya penurunan *error* (MSE) yang tidak stabil, karena grafik nilai MSE yang dihasilkan naik turun tidak terkendali. Hal ini menunjukkan bahwa dengan menggunakan *learning rate* dengan *epoch* terkecil saat pelatihan tidak menjamin nilai MSE yang dihasilkan stabil walaupun jaringan dengan cepat mencapai nilai yang konvergen. Ternyata nilai *error* (MSE) yang dihasilkan naik turun tidak terkendali. Hal ini disebabkan karena nilai *learning rate* yang terlalu besar, maka jaringan menjadi tidak stabil akibat perubahan nilai *error* dan bobot yang terlalu besar.

Grafik konvergensi pelatihan pada saat *learning rate* 1.0 yang ditunjukkan pada Gambar 5.4 menunjukkan bahwa grafik tersebut terlihat adanya penurunan *error* (MSE) yang tidak stabil, karena pada awal *epoch* pelatihan terjadi penurunan *error* (MSE) terlalu cepat dan kemudian di saat *epoch* 401 terlihat adanya peningkatan nilai MSE yang cukup drastis kemudian turun secara drastis lagi saat *epoch* 402, sehingga perubahan MSE menjadi kurang stabil. Hal ini disebabkan karena nilai *learning rate* yang terlalu besar, maka jaringan menjadi tidak stabil akibat perubahan nilai *error* dan bobot yang terlalu besar.

Grafik konvergensi pelatihan pada saat *learning rate* 0.8 yang ditunjukkan pada Gambar 5.5 menunjukkan bahwa grafik tersebut pada awal *epoch* pelatihan terlihat adanya penurunan MSE yang terlalu cepat, akan tetapi hal itu tidak berlangsung lama. Setelah itu penurunan MSE berlangsung tidak terlalu cepat maupun tidak terlalu lambat, sehingga nilai *learning rate* 0.8 inilah yang dipilih sebagai nilai parameter terbaik untuk konfigurasi jaringan optimal.

Pada dasarnya nilai *learning rate* yang terlalu besar menyebabkan jaringan tidak membutuhkan waktu yang cukup lama untuk konvergen, tetapi menghasilkan nilai perubahan *error* dan bobot yang terlalu besar. Sebaliknya, apabila nilai *learning rate* terlalu kecil, maka jaringan akan membutuhkan waktu yang lama untuk konvergen. Namun perubahan *error* dan bobot yang terjadi tetap dipengaruhi terhadap cara inisialisasi bobot dan bias awal pada saat pelatihan, karena inisialisasi inilah yang menentukan nilai *error* yang dihasilkan pada setiap *epoch* (iterasi).

5.3.3 Analisa Hasil Pengaruh Jumlah Data Latih Seimbang

Berdasarkan pada tabel percobaan pengaruh jumlah data latih seimbang yang ditunjukkan pada Tabel 5.11 dapat diketahui bahwa besarnya akurasi dipengaruhi oleh variasi jumlah data latih. Berdasarkan hasil uji coba yang telah dilakukan, terlihat bahwa jumlah data latih sangat berpengaruh terhadap besarnya nilai akurasi terlebih jika data latih dengan kelas data seimbang. Peningkatan akurasi berbanding lurus dengan penambahan jumlah data latih, hal ini dapat dilihat pada Gambar 5.6.

Penyebab nilai rata-rata akurasi yang semakin naik, karena semakin banyak variasi pola data latih yang direkam pada saat pelatihan, maka semakin baik pula JST RPROP dalam mengenali berbagai macam pola ketika pengujian, sehingga menghasilkan bobot dan bias optimal yang dapat digunakan untuk pengujian. Dalam proses pelatihan, apabila data yang dimasukkan semakin bervariasi dan dengan jumlah yang banyak, maka dengan hal ini dapat memperkaya sistem dengan berbagai macam kondisi keadaan data. JST memiliki kemampuan dalam men-generalisasi suatu ciri pola yang telah dilatih, sehingga

semakin banyak variasi ciri pola yang dilatih, maka akurasi sistem akan semakin baik.

Kesalahan prediksi pada beberapa data bisa disebabkan karena penentuan nilai *range* target pada masing-masing kelas dengan *range* keluaran masih belum sesuai, oleh karena itu menyebabkan selisih nilai antara keluaran JST (data prediksi) dengan data aktual menjadi besar. Hal ini ditunjukkan pada Gambar 5.7, yaitu nilai selisih terbesar antara data prediksi dengan data aktual 0.221942389 terdapat pada data uji ke-21 yang menyebabkan data salah menjadi terklasifikasikan.

5.3.4 Analisa Hasil Pengaruh Jumlah Data Latih Tidak Seimbang

Berdasarkan pada tabel percobaan pengaruh jumlah data latih tidak seimbang yang ditunjukkan pada Tabel 5.19 dan Gambar 5.8 dapat diketahui bahwa penambahan jumlah data latih tidak berbanding lurus dengan peningkatan nilai akurasi. Pada saat data latih tidak seimbang, terjadi dominasi data pada kelas tertentu sehingga menimbulkan *noise* yang menyebabkan terjadinya kesalahan dalam mengklasifikasikan data dan cenderung mengacu pada kelas yang ciri pola data latihnya lebih beragam.

Data latih yang didapatkan tidak sepenuhnya mengikuti aturan dari BKKBN, karena ada beberapa penilaian kriteria status tahapan keluarga sejahtera berdasarkan penilaian pakar di masing-masing instansi yang mana cara penilaian tersebut tergantung dari kasus yang sedang dihadapi. Hal ini menyebabkan nilai dari masing-masing indikator pada kelas tertentu memiliki bermacam variasi pola. Akibatnya, nilai keluaran JST tidak berada pada nilai pada *range* target yang diinginkan. Selain itu, kesalahan prediksi juga bisa disebabkan karena pemilihan nilai *range* target pada masing-masing kelas dengan *range* keluaran masih belum sesuai, oleh karena itu menyebabkan selisih nilai antara keluaran JST (data prediksi) dengan data aktual menjadi besar. Hal ini ditunjukkan pada Gambar 5.9, yaitu nilai selisih terbesar antara data prediksi dengan data aktual 0.196618053 terdapat pada data uji ke-31 yang menyebabkan data salah terklasifikasikan.

Pada saat jaringan dilatihkan dengan data latih 40%, nilai rata-rata akurasi mengalami penurunan dari 87.50% menjadi 85.67%. Kemudian mengalami sedikit peningkatan pada saat data latih 50%, yaitu rata-rata akurasi sebesar 87.67%. Pada saat dilatihkan dengan data latih 60%, nilai rata-rata akurasi mengalami peningkatan, yaitu menjadi 89.67%, lalu mengalami peningkatan lagi pada saat data latih 70%, yaitu dengan nilai rata-rata akurasi sebesar 90.00%. Berdasarkan hasil pengujian tersebut, maka dapat disimpulkan bahwa semakin banyak data latih dapat memperbaiki klasifikasi dan dapat pula merusak klasifikasi data. Jumlah distribusi data yang tidak seimbang pada masing-masing kelas dapat menyebabkan proses duplikasi akan terus dilakukan sampai data yang salah klasifikasi menjadi benar, sehingga dapat menyebabkan data yang semula sudah benar terklasifikasikan menjadi salah. Namun, secara keseluruhan dari pengujian ini dapat dikatakan bahwa dengan penambahan jumlah data latih masih menghasilkan nilai akurasi yang bagus.

BAB VI
PENUTUP

6.1 Kesimpulan

Metode jaringan syaraf tiruan (JST) dengan algoritma pembelajaran *Resilient backpropagation* dapat diimplementasikan untuk kasus klasifikasi status tahapan keluarga sejahtera dengan menggunakan konfigurasi jaringan optimal yang dihasilkan dari pelatihan. Konfigurasi jaringan optimal yang dihasilkan adalah 21 unit neuron pada *input layer*, 45 unit neuron pada *hidden layer*, dan 1 unit neuron pada *output layer* dengan kombinasi parameter terbaik dari hasil pelatihan, yaitu nilai *learning rate* sebesar 0.8.

Sistem dapat memprediksi dengan baik pada kasus klasifikasi status tahapan keluarga sejahtera dengan menggunakan jumlah data latih sebesar 70% dalam keadaan kelas seimbang *(balanced class)* dan data uji sebesar 30%, sehingga hasil prediksi dapat mencapai tingkat rata-rata akurasi dari 5 kali percobaan sebesar 90.33% dan akurasi tertinggi dari 5 kali percobaan sebesar 93.33%.

6.2 Saran

Pada penelitian jaringan syaraf tiruan dengan menggunakan metode pembelajaran *Resilient backpropagation* untuk kasus klasifikasi status tahapan keluarga sejahtera perlu dilakukan pembanding dengan menggunakan fungsi aktivasi lainnya untuk mengetahui fungsi aktivasi yang lebih optimal untuk digunakan proses pengklasifikasian kasus ini.

Selain itu, juga diperlukan pelatihan dan pengujian menggunakan 2 atau lebih *hidden layer*, sehingga diharapkan dapat diperoleh hasil prediksi klasifikasi yang lebih baik.

DAFTAR PUSTAKA

[BAS-11] Baskaraningrum, A. 2011, "Naïve Bayes Classifier untuk Pengelompokan Keluarga Sejahtera dan Keluarga Prasejahtera", Universitas Pembangunan Nasional Veteran Jakarta, Skripsi.

[DAN-12] Daniel, P. 2012, "Penerapan Metode Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) untuk Perekrutan Tenaga Kerja", Universitas Negeri Gorontalo, Skripsi.

[EFF-08] Effendy, Nazrul & kawan. 2008, "Prediksi Penyakit Jantung Koroner (PJK) Berdasarkan Faktor Risiko Menggunakan Jaringan Syaraf Tiruan Backpropagation", Seminar Nasional Aplikasi Teknologi Informasi 2008.

[FAD-09] Fadil, Jazuli. 2009, "Load Forecasting For The Distribution Neural Network Of South And Middle Kalimantan Using Artificial Neural Networks Resilient Propagation", Proceeding of National Seminar on Applied Technology, Science, And Arts 1th.

[FAU-94] Fausett, L. 1994, "Fundamentals of Neural Network: Architectures Algorithms, and Applications", Prentice-Hall, Inc, New Jersey.

[FEB-07] Febrianty, D. 2007, "Analisis Jaringan Saraf Tiruan Rprop Untuk Mengenali Pola Elektrokardiografi Dalam Mendeteksi Penyakit Jantung Koroner", Seminar Nasional Aplikasi Teknologi Informasi.

[ING-07] Ingvarson, M. 2007, "The RPROP Algorithm Resilient Propagation for NN", Materi PPT.

[JAY-12] Jaya, Tri S. 2012, "Sistem Pemilihan Perumahan dengan Metode Kombinasi Fuzzy C-Means Clustering dan Simple Additive Weighting", Universitas Diponegoro, Semarang, Tesis.

[KUS-04] Kusumadewi, S. 2003, "Artificial Inteligence (Teknik dan Aplikasinya)", Graha Ilmu, Yogyakarta.

[MAN-10] Manurung, P. 2010, "Sistem Pendukung Keputusan Seleksi Penerima Beasiswa dengan Metode AHP dan TOPSIS", Universitas Sumatera Utara, Skripsi.

[PAN-08] Panji, M. 2008, "Multifraktalitas dan Studi Komparatif Prediksi Indeks dengan Metode Arima dan Neural Network", Universitas Diponegoro, Semarang, Skripsi.

[PAN-12] Pangestuti, Nita A. 2012, "Implementasi Pelatihan Jaringan Syaraf Tiruan dengan Algoritma Genetika untuk Peramalan Cuaca", PTIIK, Universitas Brawijaya, Skripsi.

[PUS-06] Puspitaningrum, D. 2006, "Pengantar Jaringan Saraf Tiruan", Andi, Yogyakarta.

[RIE-93] Riedmiller, M. 1993, "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm", University of Karlsrube.

[SET-12] Setyosari, P. 2012, "Metode Penelitian Pendidikan dan Pengembangan", Kencana, Jakarta.

[SIA-05] Siang, Jong Jek. 2005, "Jaringan Syaraf Tiruan dan Pemrograman Menggunakan Matlab 1 edition", Andi, Yogyakarta.

[SIN-09] Sinuhaji, F. 2009, "Jaringan Syaraf Tiruan untuk Prediksi Keputusan Medis pada Penyakit Asma", Universitas Sumatera Utara, Medan, Skripsi.

[SUB-08] Subrata, I. 2008, "Peramalan Trafik Data Menggunakan Jaringan Saraf Tiruan", Universitas Diponegoro, Semarang, Makalah Seminar Tugas Akhir.

[SUB-10] Subandi dkk. 2010, "Evaluasi Pelayanan Keluarga Berencana bagi Masyarakat Miskin (Keluarga Prasejahtera/KPS dan Keluarga Sejahtera-I/KS-I", Laporan Akhir.

[SUN-07] Sunarti, E. 2007, "Indikator Keluarga Sejahtera : Sejarah Pengembangan, Evaluasi, dan Keberlanjutannya", Institut Pertanian Bogor.

[SUR-10] Suryana. 2010, "Metodologi Penelitian Model Praktis Penelitian Kuantitatif dan Kualitatif", Universitas Pendidikan Indonesia, Buku Ajar Perkuliahan.

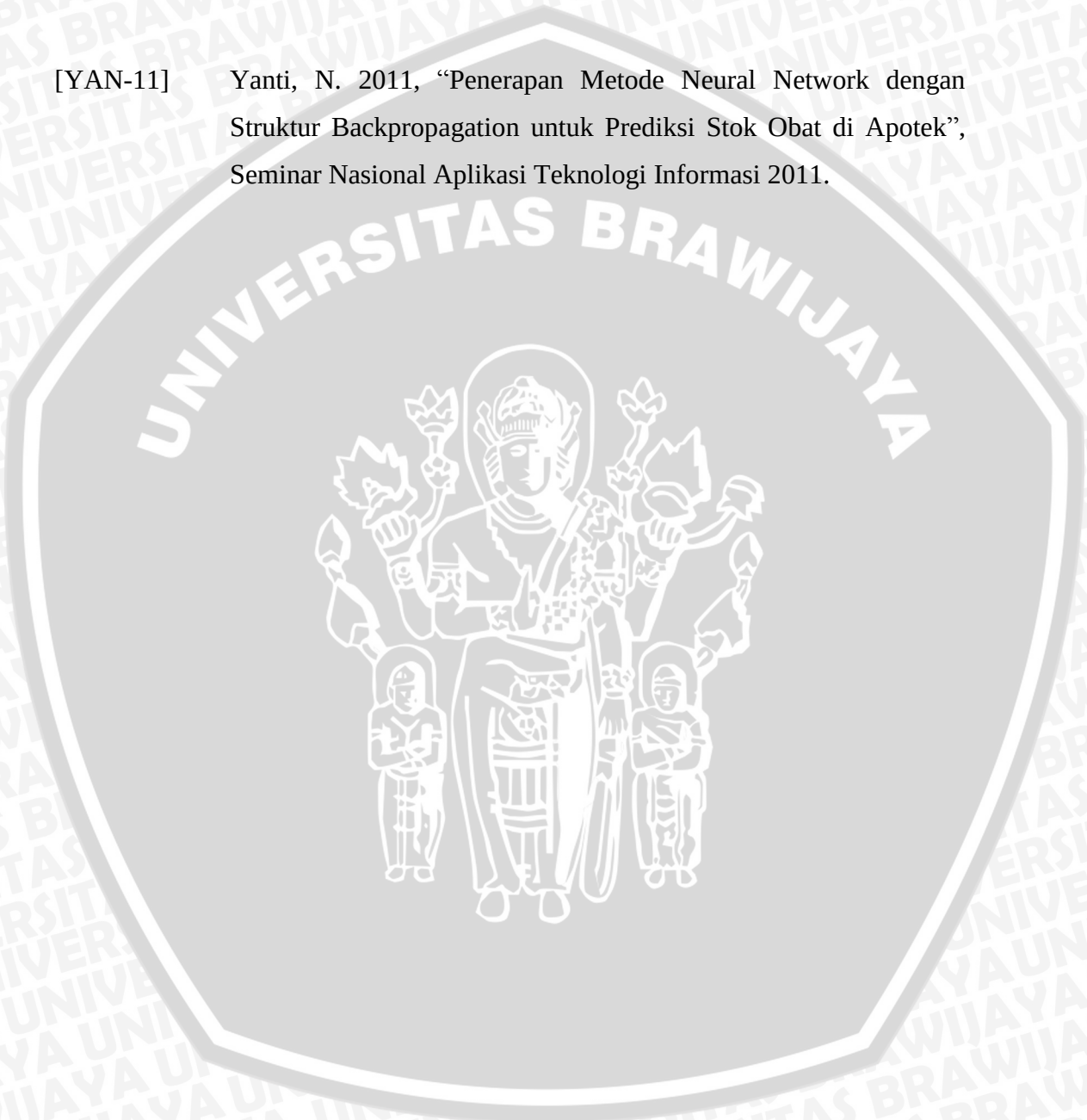
[SUS-12] Susanto, Agustin T. 2010, "Aplikasi Diagnosa Kanker Serviks dengan Menggunakan Algoritma Backpropagation", STIKOM Uyenlindo Kupang, Skripsi.

[SUP-12] Supriyadi, D. 2012, "Sistem Informasi Penyebaran Penyakit Demam Berdarah Menggunakan Metode Jaringan Syaraf Tiruan Backpropagation", Universitas Diponegoro, Semarang, Tesis.

[TAH-12] Tahir, M. 2012, "Pengantar Metodologi Penelitian Pendidikan", Universitas Muhammadiyah Makassar.

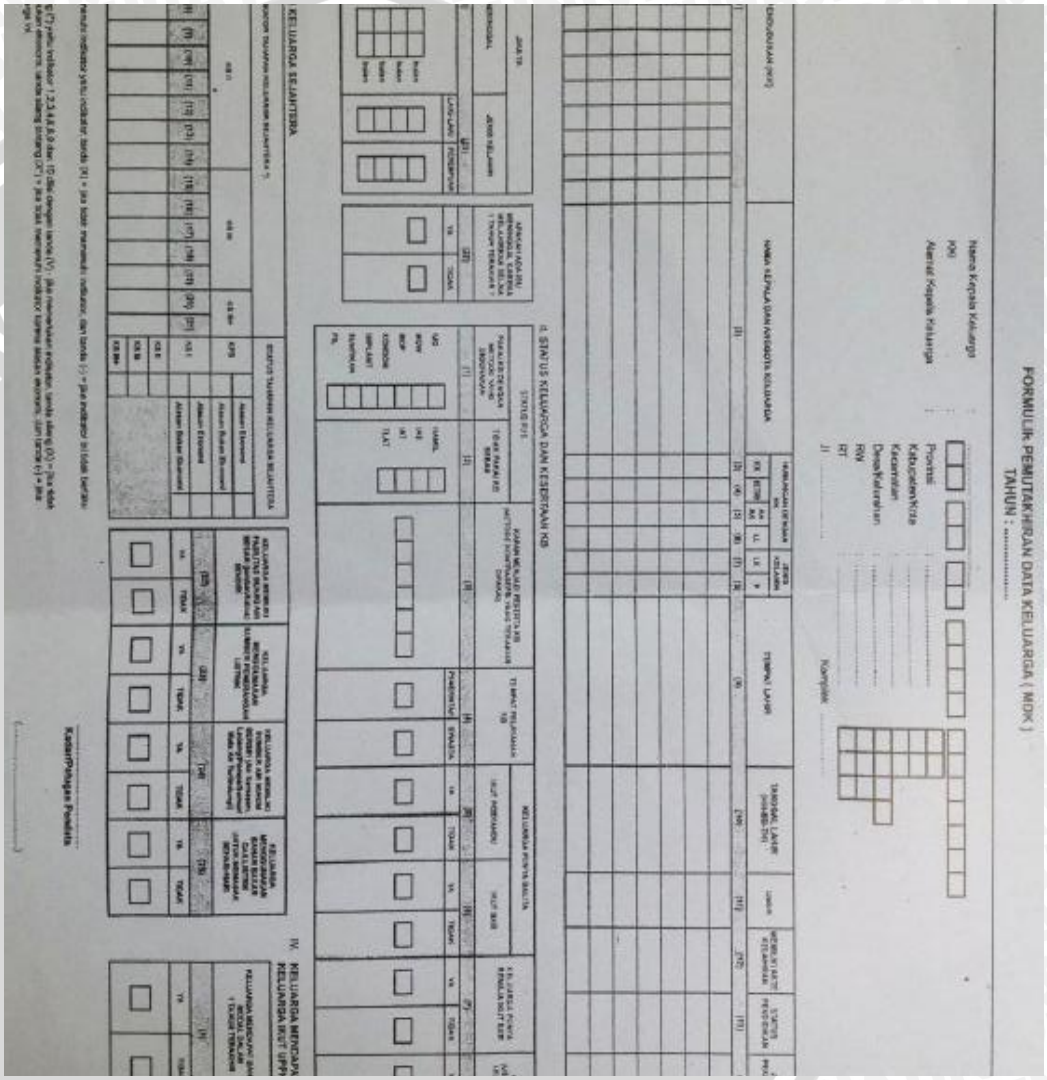
[VIR-13] Virgiandhana, S. 2013, "Penerapan Jaringan Syaraf Tiruan Resilient Backpropagation untuk Diagnosis Penyakit Diabetes Mellitus", PTIIK, Universitas Brawijaya, Jurnal.

[YAN-11] Yanti, N. 2011, "Penerapan Metode Neural Network dengan Struktur Backpropagation untuk Prediksi Stok Obat di Apotek", Seminar Nasional Aplikasi Teknologi Informasi 2011.



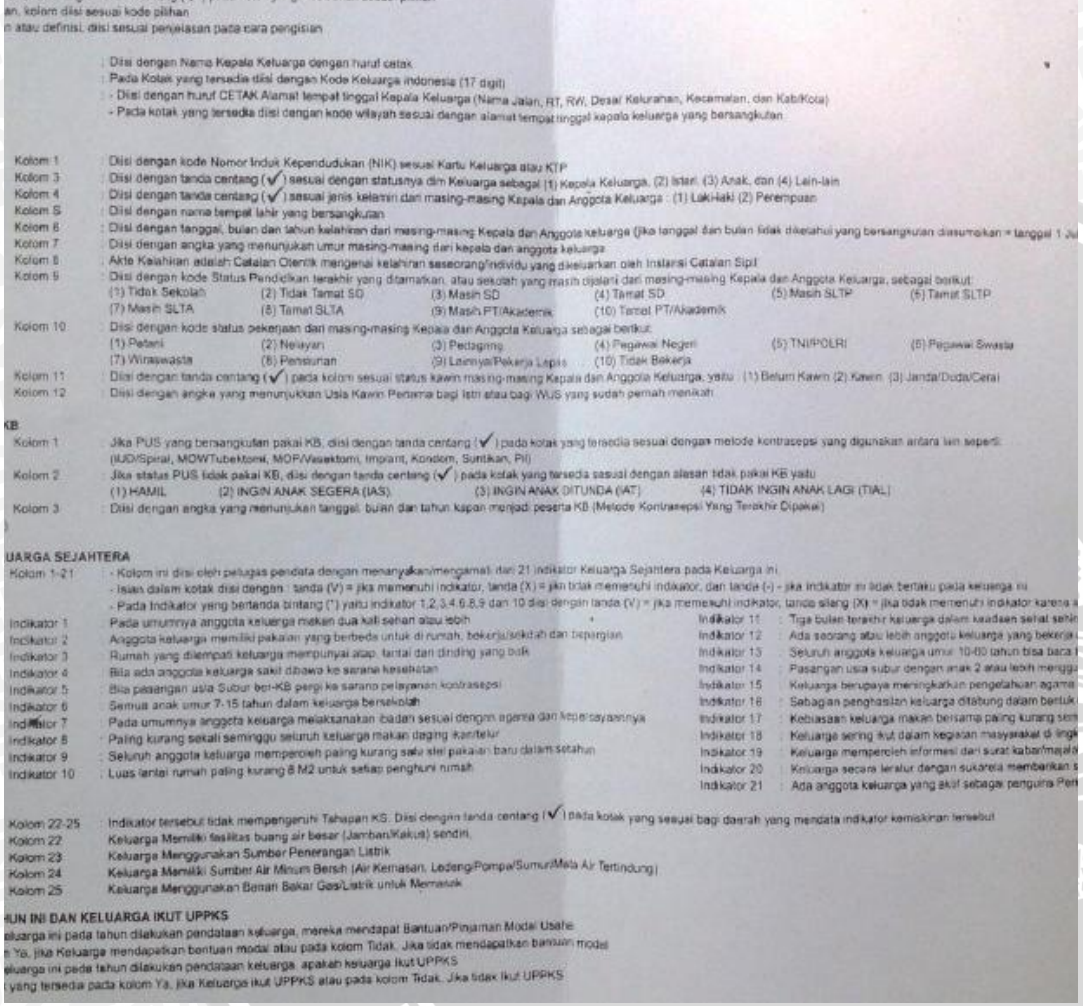
LAMPIRAN

Lampiran 1 Formulir F/I/MDK/2011 Halaman 1



L-2

Lampiran 2 Formulir F/I/MDK/2011 Halaman 2



an, kolom diisi sesuai kode pilihan
atau definisi, diisi sesuai penjelasan pada cara pengisian

: Diisi dengan Nama Kepala Keluarga dengan huruf cetak
: Pada Kotak yang tersedia diisi dengan Kode Keluarga Indonesia (17 digit)
: - Diisi dengan huruf CETAK Alamat tempat tinggal Kepala Keluarga (Nama Jalan, RT, RW, Desa/ Kelurahan, Kecamatan, dan Kab/Kota)
- Pada kotak yang tersedia diisi dengan kode wilayah sesuai dengan alamat tempat tinggal kepala keluarga yang bersangkutan

Kolom 1 : Diisi dengan kode Nomor Induk Kependudukan (NIK) sesuai Kartu Keluarga atau KTP
Kolom 3 : Diisi dengan tanda centang (✓) sesuai dengan statusnya dlm Keluarga sebagai (1) Kepala Keluarga, (2) Istri, (3) Anak, dan (4) Lain-lain
Kolom 4 : Diisi dengan tanda centang (✓) sesuai jenis kelamin dari masing-masing Kepala dan Anggota Keluarga : (1) Laki-Laki (2) Perempuan
Kolom 5 : Diisi dengan nama tempat lahir yang bersangkutan
Kolom 6 : Diisi dengan tanggal, bulan dan tahun kelahiran dari masing-masing Kepala dan Anggota keluarga (jika tanggal dan bulan tidak diketahui yang bersangkutan diasumsikan = tanggal 1 Ju
Kolom 7 : Diisi dengan angka yang menunjukan umur masing-masing dari kepala dan anggota keluarga
Kolom 8 : Akte Kelahiran adalah Catatan Otentik mengenai kelahiran seseorang/individu yang dikeluarkan oleh Instansi Catatan Sipil
Kolom 9 : Diisi dengan kode Status Pendidikan terakhir yang ditamatkan, atau sekolah yang masih dijalani dari masing-masing Kepala dan Anggota Keluarga, sebagai berikut:
(1) Tidak Sekolah (2) Tidak Tamat SD (3) Masih SD (4) Tamat SD (5) Masih SLTP (6) Tamat SLTP
(7) Masih SLTA (8) Tamat SLTA (9) Masih PT/Akademik (10) Tamat PT/Akademik
Kolom 10 : Diisi dengan kode status pekerjaan dari masing-masing Kepala dan Anggota Keluarga sebagai berikut:
(1) Petani (2) Nelayan (3) Pedagang (4) Pegawai Negeri (5) TNI/POLRI (6) Pegawai Swasta
(7) Wiraswasta (8) Pensiunan (9) Lainnya/Pekerja Lepas (10) Tidak Bekerja
Kolom 11 : Diisi dengan tanda centang (✓) pada kolom sesuai status kawin masing-masing Kepala dan Anggota Keluarga, yaitu : (1) Belum Kawin (2) Kawin (3) Janda/Duda/Cerai
Kolom 12 : Diisi dengan angka yang menunjukkan Usia Kawin Pertama bagi Istri atau bagi WUS yang sudah pernah menikah

KB

Kolom 1 : Jika PUS yang bersangkutan pakai KB, diisi dengan tanda centang (✓) pada kotak yang tersedia sesuai dengan metode kontrasepsi yang digunakan antara lain seperti: (IUD/Spiral, MOW/Tubektomi, MOP/Vasektomi, Implant, Kondom, Suntikan, Pil)
Kolom 2 : Jika status PUS tidak pakai KB, diisi dengan tanda centang (✓) pada kotak yang tersedia sesuai dengan alasan tidak pakai KB yaitu
(1) HAMIL (2) INGIN ANAK SEGERA (IAS) (3) INGIN ANAK DITUNDA (IAT) (4) TIDAK INGIN ANAK LAGI (TIAL)
Kolom 3 : Diisi dengan angka yang menunjukan tanggal, bulan dan tahun kapan menjadi peserta KB (Metode Kontrasepsi Yang Terakhir Dipakai)

UARGA SEJAHTERA

Kolom 1-21 : - Kolom ini diisi oleh petugas pendata dengan menanyakan/mengamati dari 21 indikator Keluarga Sejahtera pada Keluarga ini.
- Isian dalam kotak diisi dengan : tanda (V) = jika memenuhi indikator, tanda (X) = jika tidak memenuhi indikator, dan tanda (-) - jika indikator ini tidak berlaku pada keluarga ini
- Pada Indikator yang bertanda bintang (*) yaitu indikator 1,2,3,4,6,8,9 dan 10 diisi dengan tanda (V) = jika memenuhi indikator, tanda silang (X) = jika tidak memenuhi indikator karena a

Indikator 1 : Pada umumnya anggota keluarga makan dua kali sehari atau lebih
Indikator 2 : Anggota keluarga memiliki pakaian yang berbeda untuk di rumah, bekerja/sekolah dan bepergian
Indikator 3 : Rumah yang ditempati keluarga mempunyai atap, lantai dan dinding yang baik
Indikator 4 : Bila ada anggota keluarga sakit dibawa ke sarana kesehatan
Indikator 5 : Bila pasangan usia Subur ber-KB pergi ke sarana pelayanan kontrasepsi
Indikator 6 : Semua anak umur 7-15 tahun dalam keluarga bersekolah
Indikator 7 : Pada umumnya anggota keluarga melaksanakan ibadah sesuai dengan agama dan kepercayaannya
Indikator 8 : Paling kurang sekali seminggu seluruh keluarga makan daging /ikan/telur
Indikator 9 : Seluruh anggota keluarga memperoleh paling kurang satu stel pakaian baru dalam setahun
Indikator 10 : Luas lantai rumah paling kurang 8 M2 untuk setiap penghuni rumah

Indikator 11 : Tiga bulan terakhir keluarga dalam keadaan sehat sehi
Indikator 12 : Ada seorang atau lebih anggota keluarga yang bekerja
Indikator 13 : Seluruh anggota keluarga umur 10-60 tahun bisa baca
Indikator 14 : Pasangan usia subur dengan anak 2 atau lebih menggu
Indikator 15 : Keluarga berupaya meningkatkan pengetahuan agama
Indikator 16 : Sebagian penghasilan keluarga ditabung dalam bentuk
Indikator 17 : Kebiasaan keluarga makan bersama paling kurang sat
Indikator 18 : Keluarga sering ikut dalam kegiatan masyarakat di ling
Indikator 19 : Keluarga memperoleh informasi dari surat kabar/majala
Indikator 20 : Keluarga secara teratur dengan sukarela memberikan s
Indikator 21 : Ada anggota keluarga yang aktif sebagai pengurus Per

Kolom 22-25 : Indikator tersebut tidak mempengaruhi Tahapan KS. Diisi dengan tanda centang (✓) pada kotak yang sesuai bagi daerah yang mendata indikator kemiskinan tersebut
Kolom 22 : Keluarga Memiliki fasilitas buang air besar (Jamban/Kakus) sendiri.
Kolom 23 : Keluarga Menggunakan Sumber Penerangan Listrik
Kolom 24 : Keluarga Memiliki Sumber Air Minum Bersih (Air Kemasan, Ledeng/Pompa/Sumur/Mata Air Terlindung)
Kolom 25 : Keluarga Menggunakan Bahan Bakar Gas/Listrik untuk Memasak

UN INI DAN KELUARGA IKUT UPPKS

eluarga ini pada tahun dilakukan pendataan keluarga, mereka mendapat Bantuan/Pinjaman Modal Usaha
Ya, jika Keluarga mendapatkan bantuan modal atau pada kolom Tidak. Jika tidak mendapatkan bantuan modal
eluarga ini pada tahun dilakukan pendataan keluarga, apakah keluarga ikut UPPKS
yang tersedia pada kolom Ya, jika Keluarga ikut UPPKS atau pada kolom Tidak. Jika tidak ikut UPPKS

Lampiran 3 Data Hasil Prediksi dengan 70% Data Latih Seimbang

Data hasil prediksi ini diambil dari percobaan ke-4

Data ke-	Nilai Target Data Asli	Target Data Asli	Nilai Hasil Data Prediksi	Hasil Data Prediksi	Status	Selisih Data Prediksi dengan Data Aktual
1	0.1	Prasejahtera	0.094769226	Prasejahtera	Benar	0.005230774
2	0.1	Prasejahtera	0.108460005	Prasejahtera	Benar	0.008460005
3	0.1	Prasejahtera	0.097169396	Prasejahtera	Benar	0.002830604
4	0.1	Prasejahtera	0.107120825	Prasejahtera	Benar	0.007120825
5	0.1	Prasejahtera	0.097169396	Prasejahtera	Benar	0.002830604
6	0.1	Prasejahtera	0.097169396	Prasejahtera	Benar	0.002830604
7	0.1	Prasejahtera	0.098560232	Prasejahtera	Benar	0.001439768
8	0.1	Prasejahtera	0.1029467	Prasejahtera	Benar	0.0029467
9	0.1	Prasejahtera	0.09106024	Prasejahtera	Benar	0.00893976
10	0.1	Prasejahtera	0.097169396	Prasejahtera	Benar	0.002830604
11	0.1	Prasejahtera	0.097169396	Prasejahtera	Benar	0.002830604
12	0.1	Prasejahtera	0.083313839	Prasejahtera	Benar	0.016686161
13	0.3	KS I	0.335970325	KS I	Benar	0.035970325
14	0.3	KS I	0.297746355	KS I	Benar	0.002253645
15	0.3	KS I	0.269623265	KS I	Benar	0.030376735
16	0.3	KS I	0.311172777	KS I	Benar	0.011172777
17	0.3	KS I	0.312786596	KS I	Benar	0.012786596
18	0.3	KS I	0.240548793	KS I	Benar	0.059451207
19	0.3	KS I	0.356480076	KS I	Benar	0.056480076
20	0.3	KS I	0.299393644	KS I	Benar	0.000606356
21	0.3	KS I	0.078057611	Prasejahtera	Salah	0.221942389
22	0.3	KS I	0.302033665	KS I	Benar	0.002033665
23	0.3	KS I	0.202677842	KS I	Benar	0.097322158
24	0.3	KS I	0.335970325	KS I	Benar	0.035970325
25	0.5	KS II	0.421671153	KS II	Benar	0.078328847
26	0.5	KS II	0.403484382	KS II	Benar	0.096515618
27	0.5	KS II	0.50277194	KS II	Benar	0.00277194
28	0.5	KS II	0.436084922	KS II	Benar	0.063915078
29	0.5	KS II	0.323632797	KS I	Salah	0.176367203
30	0.5	KS II	0.50277194	KS II	Benar	0.00277194
31	0.5	KS II	0.323632797	KS I	Salah	0.176367203
32	0.5	KS II	0.487290461	KS II	Benar	0.012709539
33	0.5	KS II	0.50277194	KS II	Benar	0.00277194

Lanjutan data hasil prediksi dengan 70% data latih seimbang

Data ke-	Nilai Target Data Asli	Target Data Asli	Nilai Hasil Data Prediksi	Hasil Data Prediksi	Status	Selisih Data Prediksi dengan Data Aktual
34	0.5	KS II	0.50277194	KS II	Benar	0.00277194
35	0.5	KS II	0.368769789	KS I	Salah	0.131230211
36	0.5	KS II	0.50277194	KS II	Benar	0.00277194
37	0.7	KS III	0.697948389	KS III	Benar	0.002051611
38	0.7	KS III	0.697948389	KS III	Benar	0.002051611
39	0.7	KS III	0.697948389	KS III	Benar	0.002051611
40	0.7	KS III	0.697948389	KS III	Benar	0.002051611
41	0.7	KS III	0.697948389	KS III	Benar	0.002051611
42	0.7	KS III	0.697948389	KS III	Benar	0.002051611
43	0.7	KS III	0.697948389	KS III	Benar	0.002051611
44	0.7	KS III	0.697948389	KS III	Benar	0.002051611
45	0.7	KS III	0.697948389	KS III	Benar	0.002051611
46	0.7	KS III	0.697948389	KS III	Benar	0.002051611
47	0.7	KS III	0.668113247	KS III	Benar	0.031886753
48	0.7	KS III	0.697259624	KS III	Benar	0.002740376
49	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
50	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
51	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
52	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
53	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
54	0.9	KS III+	0.873653655	KS III+	Benar	0.026346345
55	0.9	KS III+	0.917723044	KS III+	Benar	0.017723044
56	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
57	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
58	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
59	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491
60	0.9	KS III+	0.901621491	KS III+	Benar	0.001621491

Lampiran 4 Data Hasil Prediksi dengan 70% Data Latih Tidak Seimbang

Data hasil prediksi ini diambil dari percobaan ke-2

Data ke-	Nilai Target Data Asli	Target Data Asli	Nilai Hasil Data Prediksi	Hasil Data Prediksi	Status	Selisih Data Prediksi dengan Data Aktual
1	0.1	Prasejahtera	0.072167485	Prasejahtera	Benar	0.027832515
2	0.1	Prasejahtera	0.133913383	Prasejahtera	Benar	0.033913383
3	0.1	Prasejahtera	0.102649254	Prasejahtera	Benar	0.002649254
4	0.1	Prasejahtera	0.102610618	Prasejahtera	Benar	0.002610618
5	0.1	Prasejahtera	0.102649254	Prasejahtera	Benar	0.002649254
6	0.1	Prasejahtera	0.102649254	Prasejahtera	Benar	0.002649254
7	0.1	Prasejahtera	0.101697519	Prasejahtera	Benar	0.001697519
8	0.1	Prasejahtera	0.074681876	Prasejahtera	Benar	0.025318124
9	0.1	Prasejahtera	0.053228919	Prasejahtera	Benar	0.046771081
10	0.1	Prasejahtera	0.102649254	Prasejahtera	Benar	0.002649254
11	0.1	Prasejahtera	0.102649254	Prasejahtera	Benar	0.002649254
12	0.1	Prasejahtera	0.074718189	Prasejahtera	Benar	0.025281811
13	0.3	KS I	0.313095113	KS I	Benar	0.013095113
14	0.3	KS I	0.302008983	KS I	Benar	0.002008983
15	0.3	KS I	0.293364319	KS I	Benar	0.006635681
16	0.3	KS I	0.448190191	KS II	Salah	0.148190191
17	0.3	KS I	0.295601547	KS I	Benar	0.004398453
18	0.3	KS I	0.289066578	KS I	Benar	0.010933422
19	0.3	KS I	0.378069282	KS I	Benar	0.078069282
20	0.3	KS I	0.30168213	KS I	Benar	0.00168213
21	0.3	KS I	0.111882478	Prasejahtera	Salah	0.188117522
22	0.3	KS I	0.30452646	KS I	Benar	0.00452646
23	0.3	KS I	0.394058088	KS I	Benar	0.094058088
24	0.3	KS I	0.313095113	KS I	Benar	0.013095113
25	0.5	KS II	0.468928128	KS II	Benar	0.031071872
26	0.5	KS II	0.346639342	KS I	Salah	0.153360658
27	0.5	KS II	0.504607372	KS II	Benar	0.004607372
28	0.5	KS II	0.471413208	KS II	Benar	0.028586792
29	0.5	KS II	0.303381947	KS I	Salah	0.196618053
30	0.5	KS II	0.504607372	KS II	Benar	0.004607372
31	0.5	KS II	0.303381947	KS I	Salah	0.196618053
32	0.5	KS II	0.508506641	KS II	Benar	0.008506641
33	0.5	KS II	0.504607372	KS II	Benar	0.004607372

Lanjutan data hasil prediksi dengan 70% data latih tidak seimbang

Data ke-	Nilai Target Data Asli	Target Data Asli	Nilai Hasil Data Prediksi	Hasil Data Prediksi	Status	Selisih Data Prediksi dengan Data Aktual
34	0.5	KS II	0.504607372	KS II	Benar	0.004607372
35	0.5	KS II	0.438475449	KS II	Benar	0.061524551
36	0.5	KS II	0.504607372	KS II	Benar	0.004607372
37	0.7	KS III	0.702186745	KS III	Benar	0.002186745
38	0.7	KS III	0.702186745	KS III	Benar	0.002186745
39	0.7	KS III	0.702186745	KS III	Benar	0.002186745
40	0.7	KS III	0.702186745	KS III	Benar	0.002186745
41	0.7	KS III	0.702186745	KS III	Benar	0.002186745
42	0.7	KS III	0.702186745	KS III	Benar	0.002186745
43	0.7	KS III	0.702186745	KS III	Benar	0.002186745
44	0.7	KS III	0.702186745	KS III	Benar	0.002186745
45	0.7	KS III	0.702186745	KS III	Benar	0.002186745
46	0.7	KS III	0.702186745	KS III	Benar	0.002186745
47	0.7	KS III	0.694980075	KS III	Benar	0.005019925
48	0.7	KS III	0.700845863	KS III	Benar	0.000845863
49	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
50	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
51	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
52	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
53	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
54	0.9	KS III+	0.852514779	KS III+	Benar	0.047485221
55	0.9	KS III+	0.910264655	KS III+	Benar	0.010264655
56	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
57	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
58	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
59	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596
60	0.9	KS III+	0.902204596	KS III+	Benar	0.002204596

Lampiran 5 Biodata Peneliti

BIODATA PENELITI

Nama Lengkap : Alfa Ridhoana
Nama Panggilan : Alfa
NIM : 105090613111003
Tempat/ Tanggal Lahir : Tulungagung, 2 Januari 1992
Alamat Rumah : Ds. Wonorejo RT.01/ RW. 02,
Kec. Sumbergempol,
Kab. Tulungagung, Kode Pos 66291
Alamat Kos : Jalan Sumbersari No. 88, Malang
No. Telp. : 085 746 970 327
E-mail : alfaridhoana@gmail.com
Asal Perguruan Tinggi : Universitas Brawijaya, Malang
Fakultas : Program Teknologi Informasi dan Ilmu Komputer
Jurusan : S1 Teknik Informatika/ Ilmu Komputer
Konsentrasi : Komputasi Cerdas dan Visualisasi
Riwayat Pendidikan :

- TK Dharmawanita Wonorejo, Tulungagung (1997-1998)
- SDN Wonorejo 1, Tulungagung (1998-2004)
- SMP Negeri 1 Tulungagung (2004-2007)
- SMA Negeri 1 Boyolangu, Tulungagung, Kelas bidang studi IPA (2007-2010)
- Pondok Pesantren Panggung (2007-2009)
- Mahasiswi Ilmu Komputer Universitas Brawijaya, Malang (2010-2014)
- Pesantren Luhur Malang (2010-2014)

Pengalaman Organisasi :

- Staff Divisi Infokom MOST Universitas Brawijaya (2010-2011)

Pengalaman Kepanitiaan :

- Staff Divisi Keamanan dan Kebersihan Panitia Buber HIMAMASTER 2010
- Staff Divisi Pendamping Panitia Raja Brawijaya 2011
- Staff Divisi Pendamping Panitia PK2 Maba FMIPA 2011
- Koordinator Divisi Publikasi, Dekorasi dan Dokumentasi Panitia Open Recruitment Anggota MOST 2012
- Staff Divisi Acara Lomba Cerdas Cermat Madrasah Diniyah Pesantren Luhur 2012

Pengalaman Penelitian :

No	Judul	Asal Penelitian	Tahun
1	Glide Society : Solusi Model Teknologi *Cloud Computing* untuk Pengembangan Bisnis Global	Lomba kreatif mahasiswa tingkat nasional festival ilmiah mahasiswa XII yang diadakan oleh UKM studi ilmiah mahasiswa Universitas Surakarta	2012
1	Klasifikasi Pemakaian Alat Kontrasepsi yang Potensial dengan Metode *Naïve Bayes Classifier*	Proyek penelitian BOPTN dari dosen MIPA, Universitas Brawijaya	2013
4	Klasifikasi Pemakaian Alat Kontrasepsi yang Potensial dengan Metode Jaringan Syaraf Tiruan *Backpropagation*	Proyek penelitian dari dosen MIPA, Universitas Brawijaya	2014

Lain-lain :

- Asisten praktikum Pemrograman 1 Tahun Ajaran 2010/2011
- Asisten praktikum Pemrograman 2 Tahun Ajaran 2011/2012
- Asisten praktikum Pemrograman Web Tahun Ajaran 2012/2013
- Tugas KKN-P membuat perancangan sistem informasi produksi berbasis web pada perusahaan peralatan rumah tangga di UD. Surya Malang