APLIKASI MODEL *RECURRENT NEURAL NETWORK* DAN *RECURRENT NEURO FUZZY* UNTUK PERAMALAN BANYAKNYA PENUMPANG KERETA API JABODETABEK

SKRIPSI

Diajukan Kepada Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Negeri Yogyakarta
untuk Memenuhi Sebagian Persyaratan guna Memperoleh Gelar Sarjana Sains

Oleh
Nanang Hermawan
NIM 10305141012

PROGRAM STUDI MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI YOGYAKARTA
2014

PERSETUJUAN

Skripsi yang berjudul:

APLIKASI MODEL *RECURRENT NEURAL NETWORK*
DAN *RECURREN NEURO FUZZY*
UNTUK PERAMALAN BANYAKNYA PENUMPANG KERETA API
JABODETABEK

Oleh:

Nanang Hermawan
NIM. 10305141012

Telah disetujui pada tanggal 7 Maret 2014
Untuk diujikan di hadapan dewan penguji Skripsi
Program Studi Matematika
Fakulatas Matematika dan Ilmu Pengetahuan Alam
Universitas Negeri Yogyakarta

Yogyakarta, 7 Maret 2014
Dosen Pembimbing

Dr. Dhoriva Urwatul Wutsqa
NIP. 19660331 199303 2 001

HALAMAN PENGESAHAN

Skripsi yang berjudul:

"APLIKASI MODEL *RECURRENT NEURAL NETWORK* DAN *RECURRENT NEURO FUZZY* UNTUK PERAMALAN BANYAKNYA PENUMPANG KERETA API JABODETABEK"

Yang disusun oleh:

Nama : Nanang Hermawan

NIM : 10305141012

Prodi : Matematika

Skripsi ini telah diujikan di depan Dewan Penguji Skripsi pada tanggal 14 Maret 2014 dan dinyatakan **LULUS**.

DEWAN PENGUJI

Nama	Jabatan	Tanda Tangan	Tanggal
Dr. Dhoriva U. W. 196603311993032001	Ketua Penguji	.....	19/3-2014
Musthofa,M. Sc. 198011072006041001	Sekretaris Penguji	.....	19/3-2014
Dr. Agus Maman Abadi 197008281995021001	Penguji Utama	.....	18/3-2014
Kuswari H.,M. Kom. 197604142005012002	Penguji Pendamping	.....	18/3-2014

Yogyakarta, 21 Maret 2014
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Negeri Yogyakarta
Dekan,

Dr. Hartono
NIP. 196203291987021002

HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini, saya:

Nama : Nanang Hermawan

NIM : 10305141012

Program Studi : Matematika

Fakultas : Matematika dan Ilmu Pengetahuan Alam

Judul Skripsi : APLIKASI MODEL *RECURRENT NEURAL NETWORK* DAN *RECURRENT NEURO FUZZY* UNTUK PERAMALAN BANYAKNYA PENUMPANG KERETA API JABODETABEK

Menyatakan bahwa skripsi ini benar-benar karya saya sendiri dan sepanjang pengetahuan saya, tidak terdapat karya atau pendapat yang ditulis atau diterbitkan orang lain, kecuali pada bagian-bagian tertentu yang diambil sebagai acuan atau kutipan dengan mengikuti tata penulisan karya ilmiah yang telah lazim.

Apabila ternyata terbukti pernyataan saya ini tidak benar, maka sepenuhnya menjadi tanggung jawab saya, dan saya bersedia menerima sanksi sesuai ketentuan yang berlaku.

Yogyakarta, 7 Maret 2014

Yang Menyatakan,

Nanang Hermawan
NIM 10305141012

MOTTO

“Kunci sukses adalah tekad yang besar, usaha yang keras, dan doa yang sering”

“Hidup adalah timbal balik, yang kita terima adalah apa yang kita beri”

“Percaya atas rencana yang disiapkan untuk kita, jangan pedulikan omongan orang lain”

HALAMAN PERSEMBAHAN

Syukur alhamdullillah akhirnya saya dapat menyelesaikan skripsi ini, skripsi ini saya persembahkan untuk:

Orangtua dan keluargaku yang telah mendoakan, menyemangati, serta memberikan dukungan moril maupun materil

Khoiruddin Aria Wijaya, Bayu M Iskandar, Dimas Ridwan W, Ikfan Mida N, dan Lina Febriani yang telah memberikan makna selama 4 th perkuliahan

Semua sahabat yang telah memberikan nasehat dan bersedia untuk menjadi tempat bercerita

Semua guru dan dosen yang telah memberikan ilmu sehingga saya menjadi seperti sekarang

Teman-teman yaitu Budi, Uswah, Reni, Udhi, Rian, semua teman-teman matsub 10 dan sekolah, terimakasih atas dukungan dalam menyelesaikan skripsi ini

APLIKASI MODEL *RECURRENT NEURAL NETWORK* DAN *RECURRENT NEURO FUZZY* UNTUK PERAMALAN BANYAKNYA PENUMPANG KERETA API JABODETABEK

Oleh:
Nanang Hermawan
NIM 10305141012

ABSTRAK

Recurrent neural network adalah model pada *neural network* yang mengakomodasi *output* jaringan untuk menjadi *input* jaringan kembali. *Recurrent neuro fuzzy* merupakan integrasi antara logika *fuzzy* dan *recurrent neural network*. Penelitian ini bertujuan untuk menjelaskan prosedur dan tingkat keakuratan dari pemodelan *recurrent neural network* dan *recurrent neuro fuzzy* Soegeno orde satu banyaknya penumpang kereta api Jabodetabek.

Pada dasarnya, langkah pembentukan model *recurrent neural network* dan *recurrent neuro fuzzy* yaitu pemilihan *input* jaringan berdasarkan plot ACF data, membagi data menjadi data *training* dan data *testing,* perancangan model terbaik, dan uji kesesuaian model untuk menguji layak tidaknya model digunakan untuk model peramlan. Perbedaan langkah pemodelan antara *recurrent neural network* dan *recurrent neuro fuzzy* terletak pada perancangan model terbaik. Pada *recurrent neural network*, langkah perancangan model terdiri dari mencari jumlah neuron tersembunyi dan eliminasi *input* jaringan menggunakan algoritma *backpropagation.* Sedangkan pada *recurrent neuro fuzzy*, langkah perancangan model terbaik yaitu mencari jumlah neuron lapisan tersembunyi, eliminasi *input* jaringan, pengelompokan (*cluster*ing) menjadi 3 *cluster*, dilanjutkan dengan pembelajaran jaringan syaraf menggunakan *recurrent neural network* yang berhubungan dengan anteseden (bagian *IF)* pada aturan-aturan inferensi *fuzzy* untuk mendapatkan nilai keanggotaan data pada bagian anteseden, pembelajaran jaringan syaraf yang berhubungan dengan konsekuen (bagian *THEN*) pada aturan-aturan inferensi *fuzzy,* dan penyederhanaan bagian konsekuen dengan melakukan eliminasi pada variabel *input* jaringan. Pencarian jumlah neuron tersembunyi, eliminasi *input*, serta pembelajaran menggunakan *recurrent neural network* algoritma *backpropagation.*

Peramalan banyaknya penumpang kereta api Jabodetabek ini menggunakan *input* yaitu data-data satu sampai empat bulan sebelumnya. Model terbaik dalam peramalan jumlah penumpang kereta api Jabodetabek ini adalah model *recurrent neural network*. MAPE dan MSE untuk *training* berturut-turut sebesar 1,26% dan 29500000 serta MAPE dan MSE *testing* berturut-turut 3,79% dan 2610000000. Selanjutnya model tersebut digunakan untuk meramalkan banyaknya penumpang kereta api Jabodetabek sepuluh bulan berikutnya.

Kata Kunci : *recurrent neural network, recurrent neuro fuzzy*, peramalan, penumpang

KATA PENGANTAR

Syukur alhamdullillah penulis panjatkan kepada Allah atas nikmat serta karunia yang diberikan kepada penulis untuk menyelesaikan Tugas Akhir Sripsi. Skripsi yang berjudul "Aplikasi model *recurrent neural network* dan *recurrent neuro fuzzy* untuk peramalan banyaknya penumpang kereta api Jabodetabek" disusun untuk memenuhi salah satu syarat kelulusan guna meraih gelar Sarjana Sains pada Program Studi Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Yogyakarta.

Skripsi ini tidak dapat diselesaikan tanpa bantuan, dukungan, serta bimbingan beberapa pihak. Penulis mengucapkan terimakasih kepada:

1. Bapak Dr. Hartono, M.Si, selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Yogyakarta yang telah memberikan kelancaran pelayanan dalam urusan akademik.
2. Bapak Dr. Sugiman, M.Si, selaku Ketua Jurusan Pendidikan Matematika Universitas Negeri Yogyakarta yang telah memberikan kelancaran pelayanan dalam urusan akademik.
3. Bapak Dr. Agus Maman Abadi, M.Si, selaku Ketua Program Studi Matematika Universitas Negeri Yogyakarta serta Penasehat Akademik yang telah memberikan arahan, motivasi, serta dukungan akademik kepada penulis.
4. Ibu Dr. Dhoriva U. W selaku dosen pembimbing yang telah sangat sabar memberikan bimbingan serta masukan dalam penyusunan skripsi ini.

5. Seluruh dosen Jurusan Pendidikan Matematikan Universitas Negeri Yogyakarta yang telah memberikan ilmu kepada penulis.
6. Orangtua dan keluarga yang telah memberikan doa, dukungan, serta semangat kepada penulis.
7. Sahabat-sahabat yaitu Wahyu Jati, Bayu Dwi Putra, Donafian Indratama, Hengki Adin Rivai, Taufan Budiyanto, Rohmat Fadloli, Bagas Pambudi, Alvian S. Indra, Nurfathoni, Muchlis Nurfata, dan Rahmadi Aldilah atas dorongan serta pelajaran selama ini.
8. Konco-konco "parkiran FC" serta teman-teman matematika 2010 yang telah menghibur serta menyemangati penulis.
9. Seluruh pihak yang telah memberikan dukungan, bantuan dan motivasi kepada penulis.

Penulis menyadari adanya ketidaktelitian, kekurangan dan kesalahan dalam penulisan tugas akhir skripsi ini. Oleh karena itu, penulis menerima kritik dan saran yang bersifat membangun. Semoga penulisan tugas akhir ini dapat bermanfaat bagi pembaca dan pihak yang terkait.

Yogyakarta, Maret 2014

Penulis

Nanang Hermawan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

A. Latar Belakang

Indonesia merupakan Negara kepulauan yang memiliki keragaman wilayah. Wilayah Indonesia terdiri wilayah daratan dan wilayah lautan. Pulau-pulau besar dan pulau-pulau kecil tersebar di seluruh wilayahnya serta dipisahkan oleh daerah perairan berupa samudera. Indonesia membentang 3977 mil di antara samudera Pasifik dan samudera Hindia. Luas daratan Indonesia adalah 1.922.570 km². Sedangkan luas wilayah perairannya 3.257.483 km². Kondisi wilayah Indonesia yang luas berpengaruh terhadap kondisi transportasi Indonesia itu sendiri.

Transportasi merupakan bidang yang sangat penting pengaruhnya terhadap kehidupan manusia. Kondisi wilayah Indonesia yang sangat luas serta keberagaman wilayah Indonesia menyebabkan transportasi menjadi bidang kegiatan yang sangat erat kaitannya dengan sendi-sendi kehidupan penduduk Indonesia. Masyarakat Indonesia memanfaatkan transportasi salah satunya untuk menjangkau daerah-daerah di Indonesia. Alat transportasi digunakan penduduk Indonesia sebagai alat perpindahan barang serta perpindahan mereka sendiri. Selain itu, transportasi digunakan untuk mendukung kegiatan ekonomi maupun pariwisata penduduk Indonesia.

Banyak jenis alat transportasi di Indonesia. Hal ini dilatarbelakangi dengan kondisi Indonesia itu sendiri. Alat transportasi tersebut meliputi alat transportasi darat, laut, maupun udara. Seiring dengan berkembangnya bidang transportasi Indonesia, kereta api menjadi alat transportasi yang sangat diminati. PT Kereta API Indonesia (Persero) Daerah Operasi I Jakarta adalah salah satu badan usaha milik negara yang bergerak dibidang transportasi Indonesia yaitu perkeretaapian khususnya wilayah Jakarta dan sekitarnya.

Jabodetabek adalah kawasan metropolitan yang terletak di Jakarta dan sekitarnya. Banyaknya jenis lapangan pekerjaan serta tempat-tempat tujuan pariwisata menjadikan kawasan Jabodetabek menjadi tujuan utama perpindahan penduduk di Indonesia. Di samping itu Jabodetabek juga merupakan kawasan yang menampung sebagian besar aktivitas pemerintahan, perdagangan, dan industri. Populasi kawasan Jabodetabek meningkat setiap tahunnya dikarenakan besarnya urbanisasi dari seluruh wilayah Indonesia.

Pada tahun 2010, jumlah perjalanan di Jabodetabek mencapai 21,9 juta perjalanan per hari. Sekitar 3% dari seluruh perjalanan di Jabodetabek tersebut menggunakan kereta api sebagai alat transportasi. Seiring dijadikannya kereta api sebagai alat transportasi yang diminati, banyak masalah-masalah yang muncul. Salah satu masalah tersebut adalah terjadinya penumpukan penumpang. Seringkali ditemui penumpang yang tidak dapat menggunakan kereta api karena kurangnya kapasitas kereta. Untuk mengantisipasi keadaan ini perlu dilakukan peramalan terhadap jumlah penumpang di waktu yang akan datang.

Data jumlah penumpang kereta api merupakan data bulanan, sehingga merupakan data *time series,* yaitu data yang diamati menurut urutan waktu (Montgomery et al, 2008:1). Dari data-data bulanan jumlah penumpang kereta api Jabodetabek inilah dapat dibentuk sebuah model yang dapat digunakan untuk meramalkan jumlah penumpang kereta api di masa mendatang. Peramalan pada data *time series* menggunakan data-data masa lampau untuk meramalkan keadaan di masa depan. Hasil peramalan dapat digunakan sebagai bahan pertimbangan dalam pengambilan kebijakan di masa depan. Oleh karena kemampuan untuk memperkirakan kegiatan-kegiatan atau pengambilan keputusan sangat dipengaruhi oleh tepat tidaknya hasil permalan, maka hingga sekarang terus berkembang model maupun metode peramalan pada data *time series*. Metode peramalan sangat berguna karena akan membantu dalam mengadakan pendekatan analisis terhadap pola data, sehingga dapat memberikan cara pemikiran, pengerjaan, dan memberikan ketepatan hasil ramalan yang dibuat.

Beberapa penelitian terkait dengan data jumlah penumpang kereta api telah dilakukan, diantaranya oleh Nila Widhianti (2013) yang meramalkan jumlah penumpang kereta api D.I Yogyakarta menggunakan model *RegArima* dengan variasi kalender, Eka Ferri Indayani (2009) yang meramalkan jumlah penumpang kereta api menggunakan metode Box-Jenkins. Penelitian Ispurwanto dkk (2011) yang menganalisis kepuasan penumpang gerbong kereta wanita dengan model *servqual.* Penelitian yang dilakukan Andria Prima (2011) yang meramalkan jumlah penumpang kereta api "Penataran" menggunakan ARIMA dan *eksponensial tripel winters*. Serta penelitian yang dilakukan Samuel (2011) yang meramalkan jumlah penumpang kereta api stasiun besar Medan menggunakan metode dekomposisi.

Model yang digunakan peda penelitian di atas merupakan model-model parametrik yang memerlukan asumsi seperti stasioneritas dan normalitas. Dewasa ini telah berkembang model peramalan yang lebih fleksibel, yang dikembangkan dengan menggunakan teknik penalaran yang disebut *soft computing. Soft computing* adalah salah satu teknik penalaran yang digunakan untuk pendekatan penyelesain masalah. *Soft computing* adalah teknik komputasi yang meniru akal manusia dan memiliki kemampuan untuk menalar dan belajar pada lingkungan yang penuh dengan ketidakpastian dan ketidaktepatan. Komponen utama pembentuk *soft computing* adalah *neural network* dan *fuzzy*(Kusumadewi dan Hartati, 2010:1-2).

Neural network adalah model yang terinspirasi oleh sistem saraf biologis. *Neural network* terdiri dari unsur-unsur sederhana yang beroperasi secara paralel. Seperti di alam, hubungan antara unsur-unsur sangat menentukan fungsi jaringan. Jaringan pada *neural network* dilatih untuk melakukan fungsi tertentu dengan menyesuaikan nilai-nilai dari koneksi (bobot) antar unsur-unsur. Biasanya, *neural network* disesuaikan, atau dilatih, sehingga *input* tertentu menyebabkan *output* target tertentu (Beale, et al 2010: 2).

Model recurrent neural network (RNN) adalah salah satu model *neural network*, yang merupakan pengembangan dari *feedforward neural network*. RNN adalah jaringan yang mengakomodasi *output* jaringan untuk menjadi *input* jaringan tersebut dalam rangka menghasilkan *output* jaringan berikutnya. Penelitian yang sudah dilakukan menggunakan model *recurrent neural network* antara lain penelitian yang dilakukan oleh Rosalina Berlinti (2006) yang meramalkan Indeks Harga Saham Gabungan, Rahul P Deshmukh dan AA Ghatol (2010) yang meramalkan banjir dengan model *general recurrent neural network*, Suhartono (2009) yang meramalkan permintaan listrik jangka pendek menggunakan model *Double Seasional RNN*, dan penelitian Ambar Sulistyaningsih (2013) yang memprediksi kunjungan ke candi prambanan menggunakan model RNN.

Model *fuzzy* dibangun dengan menggunakan konsep teori himpunan *fuzzy*, yang diperkenalkan oleh Lotfi A. Zadeh pada tahun 1965. Teori ini merupakan pengembangan dari teori himpunan klasik. Secara garis besar, logika *fuzzy* adalah suatu relasi yang memetakan koleksi dari objek-objek ke dalam suatu himpunan yang terletak antara 0 sampai 1. Elemen-elemen dari himpunan *fuzzy* dipetakan ke nilai keanggotaan yang bernilai 0 sampai 1 menggunakan sebuah fungsi tertentu (Ross, 2010:34).

Neural network dan logika *fuzzy* adalah model numerik yang dapat digunakan untuk melakukan peramalan. Sistem *neuro fuzzy* adalah model peramalan yang memadukan kedua jenis model tersebut. Sistem *neuro fuzzy* digunakan untuk akuisisi pengetahuan dan pembelajaran. Jaringan diinisialisasi ke dalam bentuk simbol, kemudian dilatih berdasarkan *input* dan *output* sistem. Proses inisialisasi dan pelatihan menggunakan *neural network*. Kemudian hasil pelatihan tersebut direpresentasikan dalam logika *fuzzy*.

Sistem *Neuro fuzzy* adalah model yang dapat digunakan untuk meramalkan data jumlah penumpang kereta api Jabodetabek. Penelitian-penelitian yang sudah dilakukan menggunakan metode *neuro fuzzy* antara lain: penelitian yang dilakukan oleh Agus Maman Abadi (2011) tentang peramalan suhu udara di Yogyakarta dengan model *fuzzy* yang dibentuk

dengan generalisasi metode Wang dan dipengaruhi oleh faktor perawanan dan kelembaban udara, Azriyenni dan Mustafa (2013) yang menggunakan *neuro fuzzy* pada sistem kesalahan lokasi, Hung-Wei Lin et al (2013) yang menggunakan *adaptive neuro fuzzy* sebagai kontrol dari *leader-follower mobile robots*, penelitian yang dilakukan Sri Mulyana (2007) yang memprediksi produksi benih ikan dengan logika *fuzzy*, serta penelitian Tiara Anggraeni (2013) tentang prediksi suhu udara D.I Yogyakarta dengan model *neuro fuzzy*.

Berdasarkan hasil penelitian sebelumnya dan sejauh peneliti ketahui, penggabungan *recurrent neural network* dan *neuro fuzzy* untuk peramalan *time series*, khususnya pada data jumlah penumpang kereta api Jabodetabek belum pernah dilakukan. Oleh karena itu, dalam penelitian ini akan diramalkan jumlah penumpang kereta api Jabodetabek dengan menggunakan model *RNN* dan *recurrent neuro fuzzy*. Jaringan yang digunakan pada model *RNN* adalah jaringan Elman dan sistem inferensi *neuro fuzzy* yang digunakan adalah model Soegeno orde satu. Jaringan Elman merupakan jaringan *neural network* yang mempunyai satu neuron tersembunyi dimana koneksi umpan balik terjadi pada lapisan tersembunyi kembali ke lapisan *input*. Terdapat 2 macam model sistem inferensi Soegeno, yaitu Soegeno orde nol dan Soegeno orde satu. Orde suatu model membedakan bentuk persamaan linear pada bagian konsekuen. Pada penelitian ini menggunakan model Soegeno orde satu, yaitu sebuah sistem yang mengintegrasikan antara *neural network* dan logika *fuzzy* (*fuzzy logic*) dengan bagian konsekuen pada aturan *fuzzy* berupa persamaan linear.

B. Rumusan Masalah

Berdasarkan latar belakang di atas maka rumusan masalah yang akan dibahas adalah sebagaiberikut:

1. Bagaimana prosedur pemodelan *recurrent neural network*?

2. Bagaimana prosedur pemodelan *recurrent neuro fuzzy* Soegeno orde satu?
3. Bagaimana hasil aplikasi model *RNN* dan *recurrent neuro fuzzy* Soegeno orde satu dalam peramalan jumlah penumpang kereta api Jabodetabek?

C. Tujuan Penelitian

Tujuan penulisan penelitian ini menurut rumusan masalah di atas adalah:

1. Menjelaskan prosedur pemodelan *recurrent neural network.*
2. Menjelaskan prosedur pemodelan *recurrent neuro fuzzy* Soegeno orde satu.
3. Mendeskripsikan hasil aplikasi model *RNN* dan *recurrent neuro fuzzy* Soegeno orde satu dalam peramalan jumlah penumpang kereta api Jabodetabek.

D. Manfaat Penelitian

Manfaat yang dapat diperoleh dari penelitian ini antara lain:

1. Menambah pengetahuan dan wawasan penulis tentang model untuk meramalkan jumlah penumpang kereta api Jabodetabek.
2. Dapat mengetahui ramalan jumlah penumpang kereta api.
3. Dengan peramalan jumlah penumpang, PT Kereta Api DAOP I dapat memanfaatkan penelitian ini dalam pengambilan kebijakan untuk mengatasi peningkatan jumlah penumpang, misalnya penambahan gerbong kereta jika hasil peramalan menunjukkan jumlah penumpang yang besar.
4. Penelitian ini dapat dijadikan referensi untuk penelitian selanjutnya.

BAB II
KAJIAN TEORI

Pada Bab II ini akan dibahas tentang transportasi, kereta api, konsep *time series*, autokorelasi, proses *white noise*, *neural network*, teori himpunan *fuzzy*, serta sistem *neural fuzzy*.

A. Transportasi

1. Sejarah Perkembangan Alat Transportasi

   Transportasi sudah sejak lama digunakan dalam kehidupan manusia. Sebelum tahun 1800, transportasi yang digunakan sebagai alat pengangkutan menggunakan tenaga manusia, hewan, dan sumber tenaga lain dari alam. Pengangkutan menggunakan tenaga-tenaga yang bukan mesin, maka pengangkutan dilakukan dengan sangat lama. Sekitar tahun 1860, transportasi mulai menggunakan mesin sebagai alat pengangkutan. Pada tahun tersebut, sudah dikenal alat transportasi yang dijalankan dengan tenaga mekanis seperti kapal uap dan kereta api. Pada tahun 1900-an sudah banyak berkembang alat transportasi berupa kendaraan bermotor seperti pesawat terbang dan mobil. Perkembangan alat transportasi mencapai puncaknya pada tahun 1920 sampai sekarang (Salim, 2004: 5).

2. Definisi Transportasi

   Menurut Salim (2004:60) transportasi adalah kegiatan pemindahan barang (muatan) dan penumpang dari satu tempat ke tempat lain. Dari definisi transportasi, terdapat 2 unsur penting yaitu:

   a. Pemindahan/ pergerakan
   b. Secara fisik mengubah tempat dari barang (komoditi) dan penumpang ke tempat lain.

   Transportasi biasanya sebagai dasar untuk pembangunan ekonomi serta pertumbuhan industri di masyarakat. Pertumbuhan ekonomi suatu negara sangat dipengaruhi oleh maju tidaknya transportasi di negara yang bersangkutan.

Terdapat 2 penggolongan fungsi transportasi yang berkaitan dengan kehidupan masyarakat. *Pertama* transportasi digunakan sebaga alat pengangkutan penumpang. Pengangkutan penumpang biasanya dilakukan dengan alat transportasi berupa mobil, motor serta alat transportasi pribadi lainnya. *Kedua* adalah fungsi transportasi sebagai angkutan umum. Dikenal banyak sekali angkutan umum yang ada di Indonesia contohnya pesawat terbang, bus umum, kereta api, kapal penyeberangan, dan lain sebagainya. Masyarakat nasional lebih sering menggunakan transportasi sebagai alat perpindahan barang daripada angkutan penumpang.

3. Jenis-Jenis Alat Transportasi

Transportasi mempunyai peran yang sangat besar pengaruhnya terutama dalam bidang ekonomi suatu negara. Indonesia sebagai negara berkembang mendapat pengaruh yang sangat besar dari perkembangan transportasinya. Kondisi wilayah Indonesia yang berupa daratan dan lautan yang sangat luas, menjadikan transportasi tidak dapat dipisahkan dengan segi kehidupan manusia. Kondisi wilayah Indonesia tersebut juga menyebabkan keberagaman jenis alat transportasi di Indonesia. Alat-alat transportasi tersebut antara lain:

a. Alat Transportasi Darat

Jenis-jenis alat transportasi darat antara lain: bus, truk, mobil, motor, sepeda, becak, kereta api, dan lain sebagainya.

b. Alat Transportasi Laut

Alat transportasi laut biasanya digunakan oleh masyarakat sebagai pengangkutan antar pulau atau antar daerah yang dipisahkan oleh wilayah perairan. Terdapat banyak alat pengangkutan barang maupun penumpang melalui laut. Alat transportasi tersebut antara lain: kapal *ferry*, kapal laut, kapal sungai, sampan, *gethek*, dan lain sebagainya.

c. Alat Transportasi Udara

Meskipun jenis dari alat transportasi udara tidak sebanyak alat transportasi darat dan laut, namun alat transportasi udara memegang

peranan yang sangat penting dalam bidang transportasi. Alat transportasi udara biasanya digunakan masyarakat sebagai alternatif alat transportasi untuk memperpendek waktu tempuh. Disamping itu, alat transportasi udara juga digunakan masyarakat untuk menempuh daerah-daerah yang tidak dapat ditempuh dengan alat transportasi darat dan laut seperti daerah pegunungan atau perbukitan. Jenis-jenis alat transportasi udara antara lain pesawat terbang, helikopter, helicak, dan lain sebagainya.

B. Kereta Api

Kereta api adalah sarana transportasi berupa kendaraan dengan tenaga gerak, baik berjalan sendiri maupun dirangkaikan dengan kendaraan lainnya, yang bergerak di rel. Kereta api umumnya terdiri dari lokomotif yang dikemudikan oleh tenaga manusia yang disebut masinis dengan bantuan mesin dan rangkaian kereta atau gerbong sebagai tempat pengangkutan barang dan atau penumpang. Rangkaian kereta atau gerbong tersebut berukuran relatif luas sehingga mampu memuat penumpang atau barang dalam skala yang besar. Karena sifatnya sebagai angkutan massal efektif, beberapa negara berusaha memanfaatkannya secara maksimal sebagai alat transportasi utama angkutan darat baik di dalam kota, antarkota, maupun antarnegara. Menurut Salim (2004:102) angkutan kereta api adalah penyediaan jasa-jasa transportasi di atas rel untuk membawa barang dan penumpang. Kereta api memberikan pelayanan keselamatan, nyaman, dan aman bagi penumpang.

Kereta api ditemukan pada sekitar tahun 1800 dan mengalami pekembangan sampai tahun 1860 (Salim, 2004:3). Pada mulanya dikenal kereta kuda yang hanya terdiri dari satu kereta (rangkaian). Kemudian dibuatlah kereta kuda yang menarik lebih dari satu rangkaian serta berjalan di jalur tertentu yang terbuat dari besi (rel). Kereta jenis ini yang kemudian dinamakan sepur atau yang lebih dekenal dengan kereta api.

Terdapat beberapa jenis kereta api. Jenis pertama adalah jenis kereta api menurut tenaga penggerak. Terdapat beberapa jenis kereta api menurut tenaga pengggeraknya antara lain:

1. Kereta Api Uap

   Kereta api uap adalah kereta api yang digerakkan dengan uap air yang dihasilkan dari ketel uap yang dipanaskan dengan kayu bakar, batu bara ataupun minyak bakar, oleh karena itu kendaraan ini dikatakan sebagai kereta api.

2. Kereta Api *Diesel*

   Kereta api *diesel* adalah jenis kereta api yang digerakkan dengan mesin *diesel* dan umumnya menggunakan bahan bakar mesin dari solar. Ada dua jenis utama kereta api *diesel* ini yaitu kereta api *diesel* hidrolik dan kereta api *diesel* elektrik.

3. Kereta Api Rel Listrik

   Kereta Rel Listrik, disingkat KRL, merupakan kereta rel yang bergerak dengan sistem propulsi motor listrik. Di Indonesia, kereta rel listrik terutama ditemukan di kawasan Jabotabek, dan merupakan kereta yang melayani para komuter.

4. Kereta Rel Sugeng

Jenis kedua adalah kereta api dilihat dari segi rel-nya. Jenis-jenis tersebut antara lain:

1. Kereta Api Konvensional

   Kereta api rel konvensional adalah kereta api yang biasa dijumpai. Kereta jenis ini menggunakan rel yang terdiri dari dua batang baja yang diletakan di bantalan. Di daerah tertentu yang memliki tingkat ketinggian curam, digunakan rel bergerigi yang diletakkan di tengah tengah rel tersebut serta menggunakan lokomotif khusus yang memiliki roda gigi.

2. Kereta Api *Monorel*

   Kereta api *monorel* (kereta api rel tunggal) adalah kereta api yang jalurnya tidak seperti jalur kereta yang biasa dijumpai. Rel kereta ini hanya terdiri dari satu batang besi. Letak kereta api didesain menggantung pada rel atau di atas rel. Karena efisien, biasanya digunakan sebagai alat

transportasi kota khususnya di kota-kota metropolitan dunia dan dirancang mirip seperti jalan layang.

Perkeretaapian termasuk salah satu industri pelayanan transportasi tertua di Indonesia. Pembangunan industri kereta api dimaksudkan untuk menopang kegiatan ekonomi masyarakat Indonesia. Hingga saat ini, perkeretaapian Indonesia dikelola oleh PT Kereta Api (PT KA). Penumpang kereta api mencapai 180 juta penumpang per tahun, 92% diantaranya adalah penumpang non-komersial atau kereta api kelas ekonomi (Hidayat, 2004:3). Jaringan perkeretaapian Indonesia mengelola rel sepanjang 4246 kilometer, baik untuk jaringan angkutan penumpang maupun angkutan barang. Tugas akhir ini akan membahas tentang penumpang kereta api daerah Jabodetabek, sehingga untuk selanjutnya pembahasan akan difokuskan kepada perkeretaapian Jabodetabek.

PT Kereta Api (Persero) Daerah Operasi (Daop) I Jakarta merupakan Daerah Operasi dengan wilayah yang terbentang dari Stasiun Merak di Provinsi Banten sampai dengan Stasiun Cikampek di Provinsi Jawa Barat. Kereta api pada Daop I melintasi stasiun–stasiun di wilayah Provinsi DKI Jakarta. Dengan jangkauan operasi sepanjang 608 km, daerah operasional 1 Jakarta memiliki keistimewaan dibanding daerah operasional lain. Keistimewaan tersebut diantaranya frekuensi kereta api yang padat dan volume penumpang besar. Kepadatan jumlah penumpang terkadang menyebabkan masalah yang beragam. Permasalahan tersebut seperti pedagang kaki lima, *free riders*, penumpang di atap, asongan, lonjakan penumpang, dll.

Kereta api daerah operasional (DAOP) I yang beroperasi di daerah Jabodetabek mengatur 34 kereta api yang setiap harinya berangkat dari dan menuju Jabodetabek. Dari data Wikipedia diketahui bahwa terdapat 10 stasiun-staiun yang tersebar diseluruh wilayah Jabodetabek yang berada dibawah pengoperasian DAOP I antara lain:

1. Stasiun Gambir
2. Stasiun Pasar Senen
3. Stasiun Jakarta Kota

4. Stasiun Jatinegara
5. Stasiun Tanjung Priuk
6. Stasiun Tanah Abang
7. Stasiun Manggarai
8. Stasiun Bekasi
9. Stasiun Bogor
10. Stasiun Cikampek

Saat ini, kereta api menjadi salah satu alat transportasi yang diminati oleh masyarakat. Tidak hanya biaya angkutan yang terjangkau melainkan juga waktu tempuh yang relatif singkat. Pada hari raya seperti natal dan lebaran, lonjakan penumpang kereta selalu terjadi. Tidak jarang hingga penambahan gerbong dilakukan guna memenuhi permintaan penumpang. Walaupun lonjakan ini sudah disiasati dengan penambahan gerbong, permintaan yang sangat besar dari segi penumpang menyebabkan kurang tersedianya kapasitas kereta. Banyak penumpang yang terkadang tidak dapat menikmati fasilitas kereta api. Dari latar belakang tersebut, perlu adanya suatu model yang dapat meramalkan penumpang kereta api agar penyediaan pelayangan kereta api bisa lebih efisien.

C. Konsep Dasar *Time Series*

Adanya tenggang waktu antara kejadian suatu peristiwa dan terjadinya kembali peristiwa tersebut di masa mendatang merupakan alasan utama bagi peramalan atau prediksi. Peramalan pada dasarnya adalah proses pengumpulan data-data masa lampau untuk memprediksi keadaan di masa depan. Terdapat banyak definisi tentang suatu peramalan. Peramalan adalah pengorganisasian informasi tentang fenomena-fenomena masa lalu untuk memprediksi masa depan. Fenomena tersebut berupa fakta atau kejadian yang terjadi atau dirasakan oleh salah satu indra atau pikiran (Frechtling, 2001: 8). Singgih Santoso (2009: 7) mendefinisikan peramalan sebagai upaya memperkirakan apa yang terjadi di masa depan, berbasis pada metode ilmiah serta dilakukan

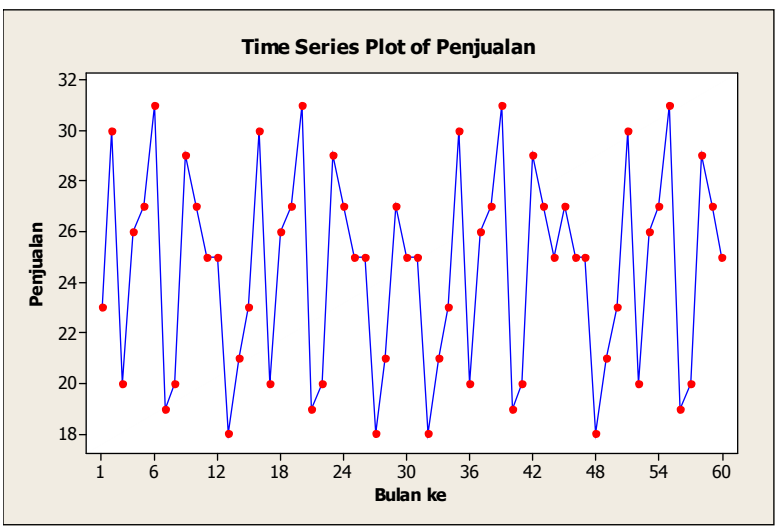
secara sistematis. Singkatnya, peramalan mengambil fakta-fakta di masa lalu dan pengetahuan ilmiah untuk membuat gambaran tentang apa yang mungkin terjadi di masa depan.

Data *time series* adalah serangkaian data mengenai nilai-nilai suatu variabel yang tersusun secara beruntun (berderet) dalam periode waktu tertentu (Montgomery et al, 2008:2). Peramalan *time series* adalah pendugaan masa depan berdasarkan nilai masa lalu dari suatu variabel (Makridakis, et al, 1999:9). Tujuan peramalan *time series* yaitu menemukan pola dalam deret data historis dan mengeksplorasikan pola tersebut ke masa depan. Langkah penting dalam menentukan metode peramalan *time series* adalah mempertimbangkan jenis pola data. Hal ini dimaksudkan agar metode yang digunakan adalah metode yang paling tepat untuk pola data yang akan diramalkan.

Terdapat empat jenis pola data (Hanke dan Wichern, 2005:58)

1. Pola Data Horisontal

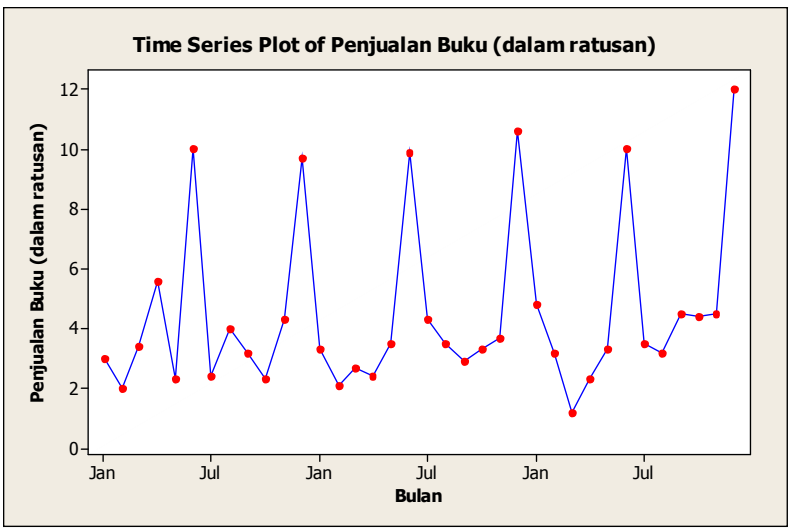
   Data dikatakan mengikuti pola horisontal jika data berfluktuasi pada suatu nilai konstan atau di sekitar rata-rata data. Tipe data berkala ini disebut dengan data stasioner. Contoh data horisontal adalah seperti pada Gambar 2.1. Dari Gambar 2.1 terlihat bahwa penjualan selalu berfluktuasi di sekitar nilai konstan tertentu.



Gambar 2.1. Contoh Plot Data Horisontal

2. Pola Data Musiman

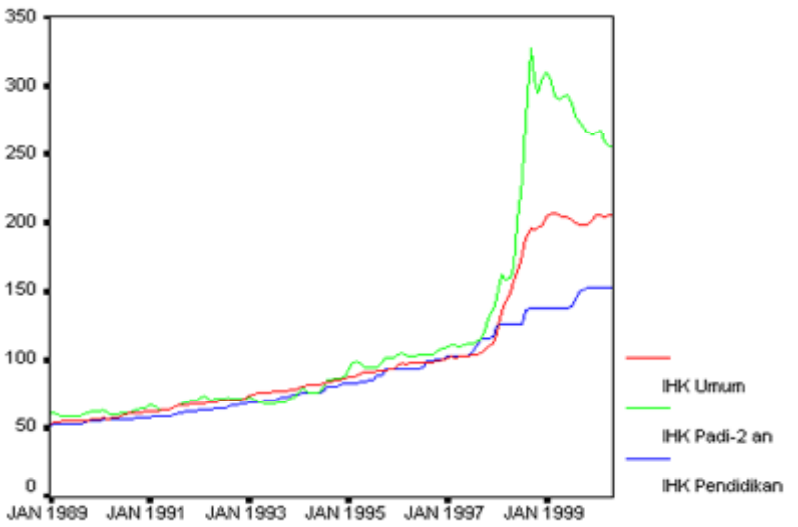
Variasi musim merupakan gerakan yang berulang-ulang secara teratur selama kurang lebih satu tahun. Pola data musiman terjadi jika data dipengaruhi oleh faktor musiman misalnya kuartalan, bulanan, atau mingguan. Contoh data musiman adalah data penjualan alat sekolah yang biasanya akan naik setiap pergantian tahun ajaran baru. Gambar 2.2 adalah plot pola data musiman



Gambar 2.2. Contoh Plot Data Musiman

3. Pola Data Siklis

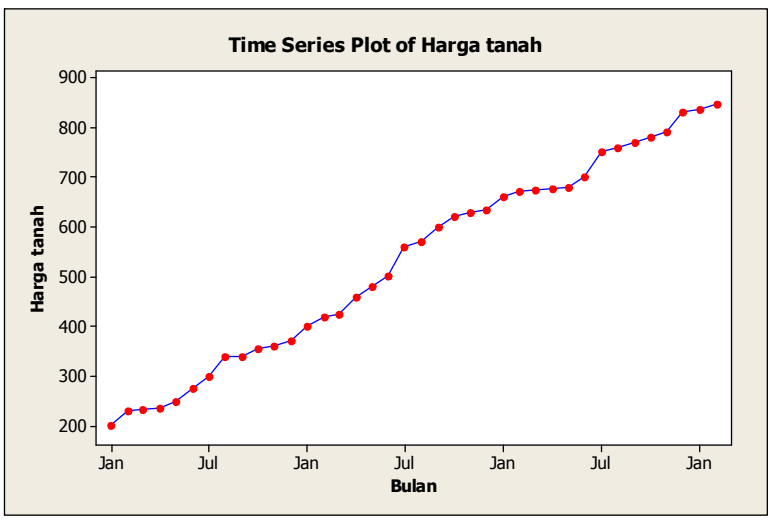
Variasi siklis adalah gerakan naik atau turun secara siklis di sekitar tren atau kondisi normal. Data yang sering mengalami gerakan siklis antara lain data perdagangan, industri, dan keuangan. Contoh data siklis adalah data IHK pada Gambar 2.3.



Gambar 2.3. Contoh Plot Data Siklis

4. Pola Data Tren

Data mengalami tren jika data tersebut bergerak pada jangka waktu tertentu dan cenderung menuju ke satu arah. Arah tersebut berupa kecenderungan naik atau turun. Contoh pola data tren seperti pada Gambar 2.4.



Gambar 2.4. Contoh Plot Data Trend

Pemilihan model adalah memilih satu atau lebih model peramalan. Baik tidaknya model didasarkan pada kriteria kebaikan model. Kriteria kebaikan model berdasarkan pada nilai *error* dari sebuah peramalan. *Error* adalah nilai

yang diperoleh dari mengurangkan nilai peramalan terhadap nilai aktual. *Error* dari sebuah peramalan dapat dihitung dengan persamaan (Hanke dan Wichern, 2005:79)

$$e_i = Y_i - \widehat{Y}_t \tag{2.1}$$

dengan

e_i : *error* peramalan pada saat t ke-*i*

Y_i : nilai aktual data ke-*i*

$\widehat{Y}_t$: nilai peramalan dari Y_i

Terdapat banyak macam kriteria kebaikan model dari sebuah peramalan. Dalam tugas akhir ini digunakan MAPE dan MSE sebagai kriteria kebaikan model. MAPE dan MSE adalah (Hanke dan Wichern, 2005:79)

1. *Mean Absolute Percentage Error (MAPE)*

   MAPE merupakan salah satu ukuran ketepatan peramalan. MAPE dapat dihitung dengan rumus

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|e_i|}{Y_i} \tag{2. 2}$$

   dengan

   MAPE : *Mean absolute percentage error*

   e_i : *error* ke-*i*

   n : banyaknya pengamatan

2. *Mean Square Error (MSE)*

   MSE adalah suatu ukuran ketepatan peramalan dengan mengkuadratkan masing-masing *error* untuk masing-masing pengamatan dalam sebuah susunan data dan kemudian memperoleh rata-rata jumlah kuadrat tersebut. *MSE* memberikan bobot yang lebih besar terhadap kesalahan yang besar dibandingkan dengan kesalahan yang kecil, sebab kesalahan dikudratkan sebelum dijumlahkan. *MSE* dapat dihitung dengan rumus

$$MSE = \sum_{i=1}^{n}\frac{{e_i}^2}{n} \tag{2.3}$$

dengan

MSE : *mean square error*

e_i : *error* ke-*i*

n : banyaknya pengamatan

Kebaikan suatu model tergantung nilai MAPE dan MSE model tersebut. Model yang baik adalah model yang memiliki MAPE dan MSE yang lebih kecil, artinya model tersebut lebih baik digunakan daripada model yang memberikan MAPE dan MSE besar.

D. Autokorelasi

Autokorelasi (*autocorrelation*) digunakan untuk menjelaskan hubungan antara data *time series* yang sama pada periode waktu yang berlainan. Autokorelasi merupakan ukuran korelasi dari sebuah data runtun waktu untuk selang waktu (*lag*) yang berlainan. Autokorelasi dapat digunakan untuk menentukan ada tidaknya faktor musiman (*seasonality*) beserta panjang musim dalam deret tersebut (Makridakis et al, 1999: 512). Selain itu, autokorelasi dapat digunakan untuk menentukan kestasioneran suatu data.

1. Fungsi Autokorelasi

Dalam suatu proses stasioner Y_t, rata-rata $E(Y_t) = \mu$ dan $var(Y_t) = E(Y_t - \mu)^2 = \sigma^2$ konstan, kovarians antara Y_t dan nilainya pada periode waktu lain Y_{t+k} disebut autokovarian pada *lag* k, didefinisikan sebagai (Wei, 2006: 10)

$$\gamma_k = Cov(Y_t, Y_{t+k}) = E(Y_t - \mu)(Y_{t+k} - \mu) \tag{2.4}$$

Nilai-nilai γ_k pada saat k = 1,2,…. disebut fungsi autokovarian. Koefisien autokorelasi pada *lag* k (ρ_k) antara pengamatan Y_t dan Y_{t+k} pada populasi dinyatakan dalam bentuk (Montgomery et al, 2008:30)

$$\rho_k = \frac{E[(Y_t-\mu)((Y_{t+k}-\mu)]}{\sqrt{E[(Y_t-\mu)^2]E[(Y_{t+k}-\mu)^2]}} = \frac{Cov(Y_t,Y_{t+k})}{var(Y_t)} = \frac{\gamma_k}{\gamma_0} \tag{2.5}$$

dimana $var(Y_t) = var(Y_{t+k}) = \gamma_0$

dengan

γ_k : fungsi kovarians pada lag-*k*

ρ_k : fungsi autokorelasi pada lag-*k*

t : waktu pengamatan, *t* = 1,2,3,…

Y_t : pengamatan pada saat *t*

Y_{t+k} : pengamatan pada saat *t* + *k*

Nilai-nilai ρ_k pada saat *k* =1, 2, ….. disebut fungsi autokorelasi (*Autocorrelation Function*/ ACF). Fungsi autokorelasi dapat diperkirakan dari fungsi autokorelasi sampel yang didefinisikan dengan (Montgomery et al, 2008:30)

$$r_k = \frac{c_k}{c_0} \tag{2.6}$$

dengan c_k adalah perkiraan fungsi autokovarian sampel yang didefinisikan sebagai

$$c_k = \frac{1}{T}\sum_{t=1}^{T-k}(Y_t - \bar{Y})((Y_{t+k} - \bar{Y}) \tag{2.7}$$

dengan

r_k : autokorelasi pada lag *k*

Y_t : pengamatan pada saat *t*

Y_{t+k} : pengamatan pada saat *t* + *k*

$\bar{Y}$: nilai rata–rata dari pengamatan

Nilai autokorelasi berkisar antar -1 sampai 1. Jika nilai autokorelasi tepat ± 1 atau mendekati, dapat disimpulkan terdapat hubungan yang tinggi antara data *time series* tersebut dalam *lag* yang berlainan. Jika nilai autokorelasi adalah 0, maka tidak terdapat hubungan dari data *time series* tersebut. Untuk mengetahui suatu autokorelasi signifikan atau tidak dapat menggunakan suatu pengujian dengan hipotesis sebagai berikut:

$H_0 : \rho_k = 0$ (koefisien autokorelasi *lag k* tidak signifikan)

$H_1 : \rho_k \neq 0$ (koefisien autokorelasi *lag k* signifikan)

Statistik uji yang digunakan adalah

$$t_{hitung} = \frac{r_k}{SE(r_k)} \tag{2.8}$$

dengan SE adalah standart *error* yang didefinisikan (Hanke dan Wichern, 2005: 64)

$$SE(r_k) = \sqrt{\frac{1+2\sum_{i=1}^{k-1} r_i^2}{n}} \tag{2.9}$$

dengan

$SE(r_k)$: standar *error* koefisien autokorelasi pada *lag k*

r_k : koefisien autokorelasi pada *lag k*

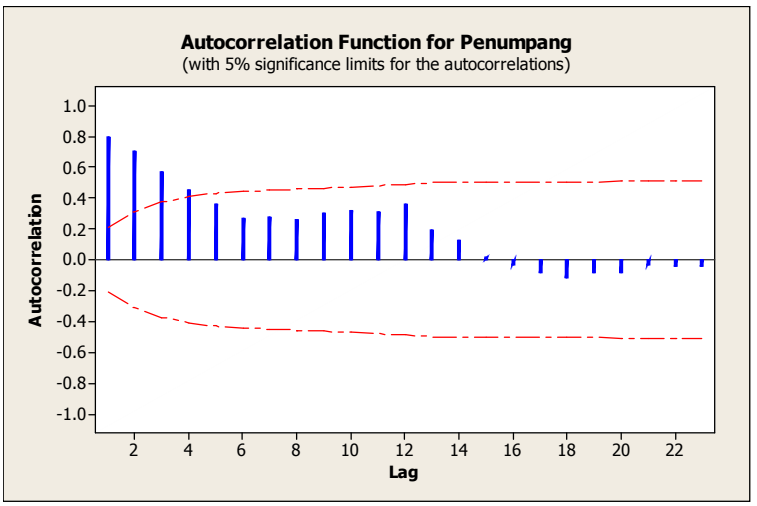
n : banyak pengamatan.

Kriteria keputusan dari pengujian ini adalah autokorelasi signifikan jika $|t_{hitung}| > t_{\frac{\alpha}{2}}$ dengan derajat bebas *n*-1.

Signifikansi autokorelasi juga dapat dilihat dengan selang kepercayaan r_k dengan pusat 0. Selang kepercayaan r_k dapat dihitung dengan rumus

$$0 \pm t_{n-1}(\alpha/2) \times SE(r_k) \tag{2.10}$$

Selang kepercayaan r_k dapat direpresentasikan dalam sebuah plot autokorelasi dengan bantuan program Minitab 16. Contoh plot autokorelasi dapat dilihat dari Gambar 2.5. Selang kepercayaan direpresentasikan dengan garis putus-putus merah. Kriteria dari suatu *lag* dikatakan signifikan jika autokorelasi melewati garis putus-putus merah. Pada Gambar 2.5. *lag* yang signifikan adalah *lag* 1 sampai dengan *lag* 4.



Gambar 2.5. Contoh Plot Autokorelasi

2. Fungsi Autokerelasi Parsial

Autokorelasi parsial *atau partial autocorrelation* (PACF) adalah ukuran korelasi antara suatu data *time series* pada saat ini dengan nilai-nilai sebelumnya dengan menganggap nilai setelahnya adalah konstan (Makridakis et al, 1999:526). Dengan kata lain, PACF digunakan untuk mengukur korelasi Y_t dengan Y_{t+k} dengan mengabaikan nilai-nilai setelah Y_{t+k}. Autokorelasi parsial biasa didefinisikan dengan

$$Corr(Y_t, Y_{t+k} | Y_{t+1}, \ldots, Y_{t+k-1}) \quad (2.11)$$

Misalkan Y_t adalah proses dengan asumsi $E(Y_t) = 0$, selanjutnya Y_{t+k} dapat dinyatakan sebagai model linear (Wei, 2006:12)

$$Y_{t+k} = \phi_{k1} Y_{t+k-1} + \phi_{k2} Y_{t+k-2} + \cdots + \phi_{kk} Y_t + e_{t+k} \quad (2.12)$$

dimana ϕ_{ki} adalah parameter ke-i dari persamaan regresi dan e_{t+k} adalah *error* berdistribusi normal yang tidak berkorelasi dengan Y_{t+k-j} untuk $j=1,2,\ldots,k$. Dengan mengalikan Y_{t+k-j} di kedua sisi dari Persamaan (2.12), didapatkan

$$Y_{t+k-j} Y_{t+k} = Y_{t+k-j} \phi_{k1} Y_{t+k-1} + Y_{t+k-j} \phi_{k2} Y_{t+k-2} + \cdots + Y_{t+k-j} \phi_{kk} Y_t + Y_{t+k-j} e_{t+k} \quad (2.13)$$

Dengan mengambil nilai harapan dari Persamaan (2.13), diperoleh persamaan

$$E(Y_{t+k-j}Y_{t+k}) = E(Y_{t+k-j}\phi_{k1}Y_{t+k-1}) + E(Y_{t+k-j}\phi_{k2}Y_{t+k-2}) + \cdots + E(Y_{t+k-j}\phi_{kk}Y_t) + E(Y_{t+k-j}e_{t+k}) \quad (2.14)$$

Menggunakan definisi autokovarian pada Persamaan (2.4) maka diperoleh

$$\gamma_j = \phi_{k1}\gamma_{j-1} + \phi_{k2}\gamma_{j-2} + \cdots + \phi_{kk}\gamma_{j-k} \quad (2.15)$$

Jika kedua ruas pada Persamaan (2.15) dibagi dengan $\gamma 0$, maka berdasarkan definisi pada persamaan (2.5) diperoleh

$$\rho_j = \phi_{k1}\rho_{j-1} + \phi_{k2}\rho_{j-2} + \cdots + \phi_{kk}\rho_{j-k} \quad (2.16)$$

Untuk *j*=1,2,…,*k*. Dengan mensubtitusikan nilai-nilai *j* didapat Persamaan (2.17) sebagai berikut

$$\rho_1 = \phi_{k1}\rho_0 + \phi_{k2}\rho_{-1} + \cdots + \phi_{kk}\rho_{1-k}$$
$$\rho_2 = \phi_{k1}\rho_1 + \phi_{k2}\rho_0 + \cdots + \phi_{kk}\rho_{2-k}$$
$$\vdots$$
$$\rho_k = \phi_{k1}\rho_{k-1} + \phi_{k2}\rho_{k-2} + \cdots + \phi_{kk}\rho_0$$

karena untuk $\rho_k = \rho(t, t+k) = \rho(t-k, t) = \rho_{-k}$, Persamaan (2.17) menjadi Persamaan (2.18)

$$\rho_1 = \phi_{k1}\rho_0 + \phi_{k2}\rho_1 + \cdots + \phi_{kk}\rho_{k-1}$$
$$\rho_2 = \phi_{k1}\rho_1 + \phi_{k2}\rho_0 + \cdots + \phi_{kk}\rho_{k-2}$$
$$\vdots$$
$$\rho_k = \phi_{k1}\rho_{k-1} + \phi_{k2}\rho_{k-2} + \cdots + \phi_{kk}\rho_0$$

Menggunakan aturan *Cramer's* untuk *k*=1, 2,….didapatkan

$$\phi_{11} = \rho_1 \quad (2.19)$$

$$\phi_{22} = \frac{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & \rho_2 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{vmatrix}} \quad (2.20)$$

$$\phi_{33} = \frac{\begin{vmatrix} 1 & \rho_1 & \rho_1 \\ \rho_1 & 1 & \rho_2 \\ \rho_2 & \rho_1 & \rho_3 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 \\ \rho_1 & 1 & \rho_1 \\ \rho_2 & \rho_1 & 1 \end{vmatrix}} \quad (2.21)$$

dan seterusnya. Sehingga koefisien autokorelasi parsial yang dinotasikan dengan ϕ_{kk} dan dirumuskan (Wei, 2006:13)

$$\phi_{kk} = \frac{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_1 \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_{1\mu} & 1 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-1} & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_{k-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & 1 \end{vmatrix}} \tag{2.22}$$

Untuk mengetahui suatu autokorelasi parsial signifikan atau tidak dapat menggunakan suatu pengujian dengan hipotesis sebagai berikut:

$H_0 : \phi_{kk} = 0$ (koefisien autokorelasi parsial *lag k* tidak signifikan)

$H_1 : \phi_{kk} \neq 0$ (koefisien autokorelasi parsial *lag k* signifikan)

Statistik uji yang digunakan adalah

$$t_{hitung} = \frac{\phi_{kk}}{SE(\phi_{kk})} \tag{2.23}$$

dengan SE adalah standart *error* yang didefinisikan (Wei, 1994: 22):

$$SE(\hat{\phi}_{kk}) = \sqrt{\frac{1}{n}} \tag{2.24}$$

dengan

$SE(\hat{\phi}_{kk})$: standar *error* koefisien autokorelasi parsial pada *lag k*

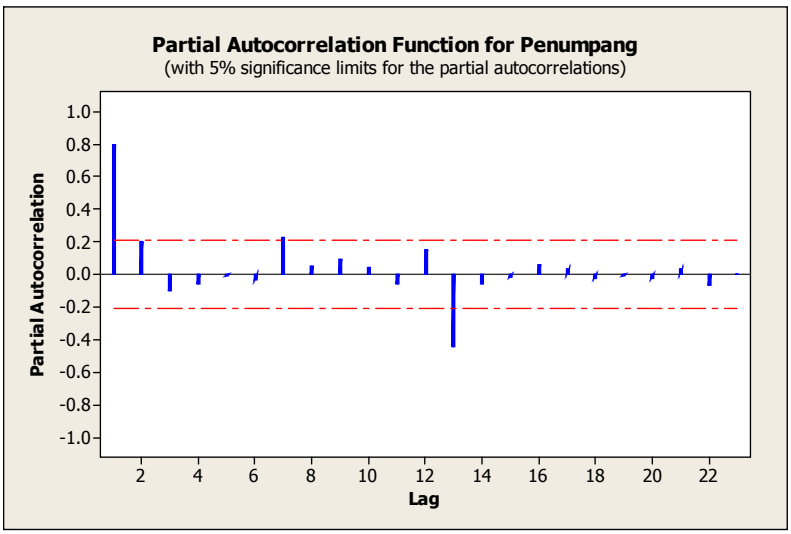
ϕ_{kk} : koefisien autokorelasi parsial pada *lag k*

n : banyak pengamatan.

Kriteria keputusan dari pengujian ini adalah autokorelasi signifikan jika $|t_{hitung}| > t\frac{\alpha}{2}$ dengan derajat bebas *n*-1.

Seperti halnya autokorelasi, signifikansi autokorelasi parsial juga dapat dilihat dengan selang kepercayaan. Contoh plot autokorelasi parsial dapat

dilihat dari Gambar 2.6. Selang kepercayaan direpresentasikan dengan garis putus-putus merah. Kriteria dari suatu *lag* dikatakan signifikan jika autokorelasi parsial melewati garis putus-putus merah. Pada Gambar 2.6 *lag* yang signifikan adalah *lag* 1, *lag* 7 dan *lag* 13.



Gambar 2.6. Contoh Plot Autokorelasi Parsial

E. Proses *White Noise*

Proses *white noise* $\{a_t\}$ adalah barisan variabel acak yang tidak berkorelasi dan berdistribusi dengan rata-rata konstan yaitu $E(a_t)=\mu_a = 0$, variansi konstan yaitu $\text{Var}(a_t)=\sigma_a^2$ dan $\gamma_k=\text{Cov}(a_t, a_{t+k}) = 0$ untuk $k \neq 0$ (Wei, 2006: 15-16). Proses *white noise* $\{a_t\}$ adalah stasioner dengan fungsi autokovarian

$$\gamma_k = \begin{cases} \sigma_a^2, & k = 0 \\ 0, & k \neq 0 \end{cases} \tag{2.25}$$

dengan fungsi autokorelasi

$$\rho_k = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases} \tag{2.26}$$

dan fungsi autokorelasi parsial

$$\phi_{kk} = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases} \quad (2.27)$$

Karena $\rho_0 = \phi_{00} = 1$, maka pada proses *white noise*, autokorelasi tidak berbeda signifikian dari nol. Untuk menguji proses *white noise* pada *error*, digunakana hipotesis sebagai berikut

H_0: $\rho_1 = \cdots = \rho_m = 0$ (autokorelasi *error* tidak signifikan artinya *error white noise*)

H_1: $\rho_k \neq 0, k = 1,2, \ldots m$ (autokorelasi *error* signifikan artinya *error* tidak *white noise*)

Statistik uji yang digunakan adalah *Ljung-Box-Pierce* (Wei, 2006: 153)

$$Q_{hitung} = n(n+2) \sum_{k=1}^{m} \frac{\hat{\rho}_k^2}{n-k} \quad (2.28)$$

dengan

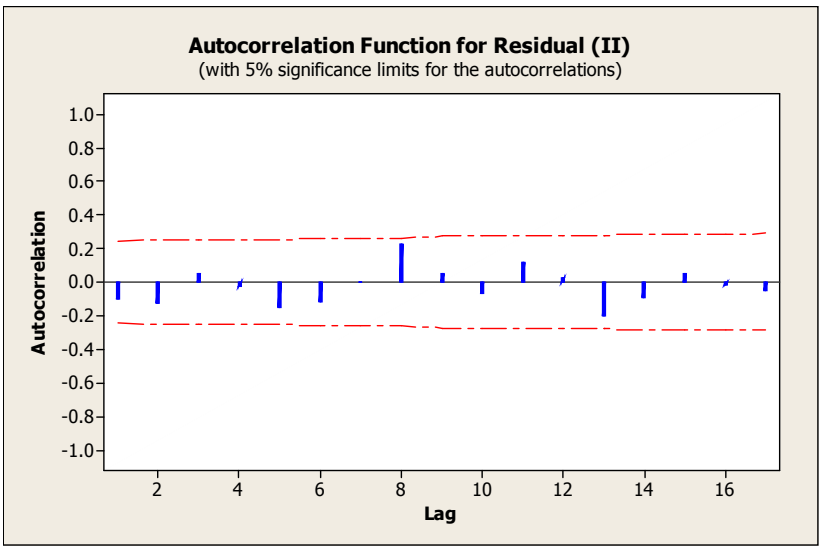
n : banyaknya pengamatan

m : banyaknya *lag* yang diuji

$\hat{\rho}_k$: perkiraan autokorelasi *lag k*

Kriteria keputusan yaitu H_0 ditolak jika $Q_{hitung} > \chi^2_{\alpha,df}$ tabel dengan *df* adalah m dikurangi banyak parameter.

Alternatif lain pengujian *white noise error* adalah dengan selang kepercayaan autokorelasi dan autokorelasi parsial yang berpusat di 0. Kriteria *error white noise* jika tidak ada *lag* yang melewati garis putus-putus merah. Seperti terlihat pada Gambar 2.7. Karena tidak ada *lag* yang melewati garis, maka *error white noise*.



Gambar 2.7. Contoh Plot ACF *Error*

F. *Neural Network*

1. Konsep Dasar *Neural Network*

Neural network atau jaringan syaraf tiruan sudah berkembang sejak tahun 1943 ketika Mc Culloch seorang ahli neuro biologi dan Pitts seorang ahli statistika menerbitkan makalah yang berjudul "*A Logical Calculus of Ideas Imminent in Nervous Activity*" di salah satu buletin matematika. Makalah tersebut terinspirasi dari perkembangan komputer digital modern. Pada saat yang sama, Frank Rosenblatt terinspirasi dari makalah ini dan kemudian melakukan suatu penelitian yang menyebabkan tercetusnya pengetahuan mengenai *neural network* genereasi pertama yaitu perceptron (Hu & Hwang. 2002:13).

Jaringan Syaraf tiruan atau yang lebih dikenal dengan *Artificial Neural Network* adalah sebuah model yang terdiri dari elemen-elemen pengolahan serta beberapa neuron atau node yang berfungsi berdasarkan cara kerja neuron pada manusia (Gurney, 1997:1). Pembuatan struktur jaringan *neural network* terinspirasi dari struktur jaringan syaraf pada manusia. Seperti halnya syaraf manusia, *neural network* terbentuk dari struktur dasar neuron yang terhubung

antara satu dan yang lain. Neuron-neuron pada *neural network* sebagai elemen pemroses yang kemudian berfungsi sebagai elemen pengahasil *output*.

Pada dasarnya *neural network* seperti metode pendekatan lainnya yang menghubungkan antara variabel-variabel *input* dengan satu atau lebih variabel *output*. Perbedaannya dengan metode pendekatan lainnya yaitu adanya satu atau lebih lapisan tersembunyi yang menghubungkan lapisan *input* dengan lapisan *output* dan ditransformasi menggunakan fungsi aktivasi. Struktur *neural network* seperti yang ditunjukkan pada gambar 2.8. Lapisan-lapisan penyusun *neural network* dibagi menjadi 3 yaitu (Siang, 2005: 24)

a. Lapisan *input*

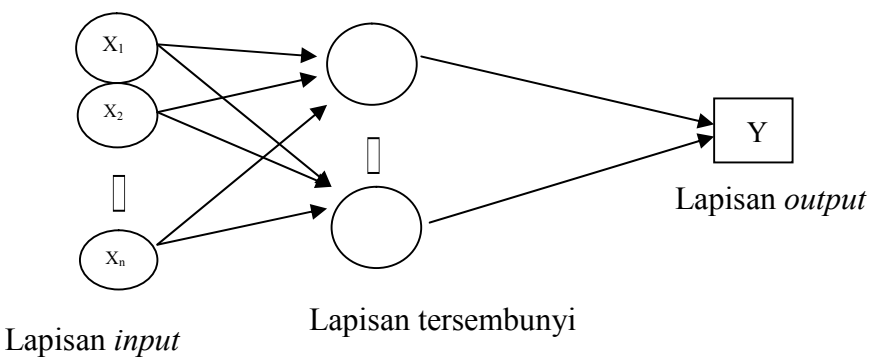
   Neuron di dalam lapisan *input* disebut neuron *input*. Neuron *input* menerima *input* dari luar, *input* yang diberikan merupakan penggambaran suatu permasalahan.

b. Lapisan tersembunyi

   Neuron di dalam lapisan tersembunyi disebut neuron tersembunyi. *Output* dari lapisan ini tidak dapat diamati secara langsung.

c. Lapisan *output*

   Neuron di dalam lapisan *output* disebut neuron *output*. Keluaran dari lapisan ini merupakan hasil dari *neural network* terhadap suatu permasalahan.



Gambar 2.8. Struktur Dasar *Neural Network*

Neuron-neuron dalam *neural network* berfungsi sebagai elemen pemroses yang berfungsi seperti halnya sebuah neuron pada hewan. Sejumlah sinyal masukan dikalikan dengan masing-masing bobot yang bersesuaian. Kemudian dilakukan penjumlahan dari seluruh hasil perkalian tersebut dan keluaran yang dihasilkan digunakan untuk mendapatkan *output* jaringan.

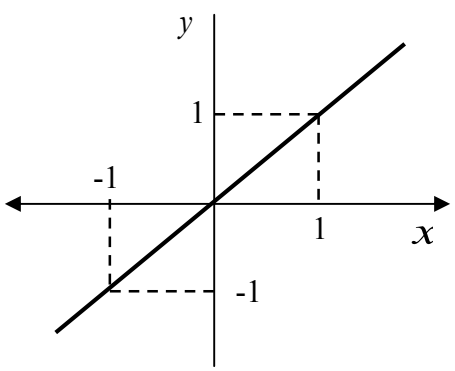
Mengaktifkan *neural network* sama halnya mengaktifkan setiap neuron yang dipakai dalam jaringan tersebut. Untuk mengaktifkan neuron, digunakan fungsi tertentu. Fungsi-fungsi tersebut disebut dengan fungsi aktivasi. Menurut Fauset (1994: 17), fungsi aktivasi akan menentukan *output* suatu neuron yang akan dikirim ke neuron lain. Fungsi aktivasi berfungsi untuk mengubah sinyal *output* menjadi sinyal *input*.

Terdapat beberapa fungsi aktivasi yang sering digunakan dalam pemodelan *neural network*. Fungsi aktivasi yang disediakan pada *toolbox* Matlab beserta perintahnya pada matlab antara lain (Kusumadewi, 2004: 51) Fungsi undak biner (*hardlim*), fungsi bipolar (*hardlims*), fungsi linear (*purelin*), fungsi *saturating* linear (*satlin*), fungsi *symmetric* saturating linear (*satlins*), fungsi sigmoid biner (*logsig*), dan fungsi sigmoid bipolar (*tansig*). Dalam tugas akhir ini akan digunakan fungsi yang dapat didiferensialkan, yaitu fungsi linear dan sigmoid bipolar sebagai fungsi aktivasi model.

a. Fungsi Linear (Identitas)

Fungsi Linear atau identitas karena memiliki nilai *output* yang sama dengan nilai *inputnya* (Zilouchian, 2001:4). Fungsi linear dirumuskan sebagai:

$$y = x \tag{2.29}$$



Gambar 2.9. Fungsi Linear

Pada matlab, fungsi aktivasi linear dikenal dengan nama *purelin* dan perintah fungsi tersebut adalah

Y=purelin(a)

a. Fungsi Sigmoid Bipolar

Fungsi sigmoid bipolar ini adalah pengembangan fungsi sigmoid biner, hanya saja *output* dari fungsi sigmoid bipolar memiliki range antara -1 sampai1. Menurut Lin & Lee (1996:209), fungsi sigmoid bipolar dirumuskan sebagai:

$$y = f\ x\ = \frac{2}{1+e^{-x}} - 1 \tag{2.30}$$

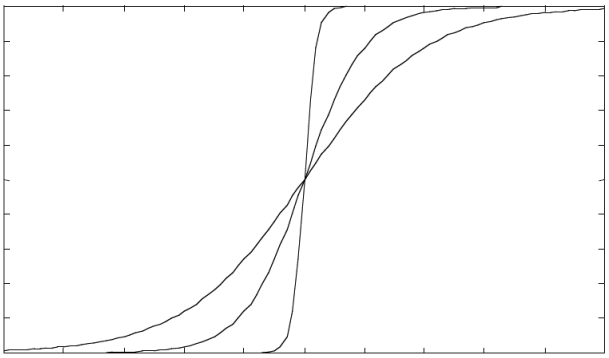
yang kemudian disederhanakan menjadi

$$y = f\ x\ = \frac{1-e^{-x}}{1+e^{-x}} \tag{2.31}$$

dengan turunan fungsi tersebut adalah:

$$f'(x) = \frac{1}{2}\ 1+ f(x)\ \ 1- f(x) \tag{2.32}$$

Gambar 2.10 adalah gambar fungsi aktivasi sigmoid bipolar



Gambar 2.10. Fungsi Aktivasi Sigmoid Bipolar

Pada toolbox *matlab*, fungsi aktivasi sigmoid bipolar dikenal dengan nama *tansig* dan dijalankan dengan perintah

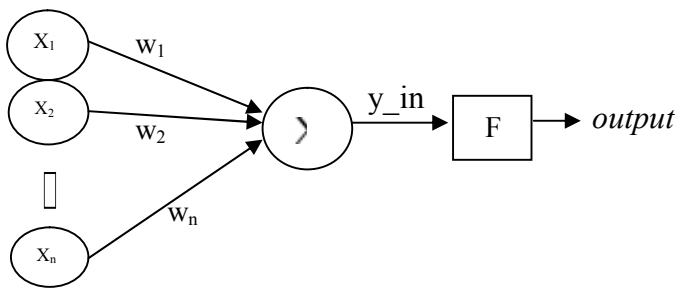
Y = tansig(a)

2. Aristektur *Artificial Neural Network*

Pada dasarnya model *neural network* terdiri dari 3 lapisan yaitu lapisan *input*, lapisan tersembunyi, dan lapisan *output*. Jumlah lapisan pada *neural network* akan mempengaruhi arsitektur *neural network* itu sendiri. Kemudian hubungan antar *neuron* dalam jaringan mengikuti pola tertentu tergantung arsitektur *neural network.* Menurut Fausset (1994: 12-15) terdapat 3 arsitektur *neural network* antara lain:

a. *Neural Network* Lapisan Tunggal (*Single Layer Net*)

Neural network dengan lapisan tunggal hanya memiliki satu lapisan dengan bobot-bobot terhubung. Jaringan pada model ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi. Dengan kata lain, jaringan ini adalah jaringan tanpa lapisan tersembunyi. Ciri-ciri arsitektur jaringan ini adalah hanya terdiri dari satu lapisan *input* dan satu lapisan *output*. Gambar 2.11 adalah contoh *neural network* lapisan tunggal.

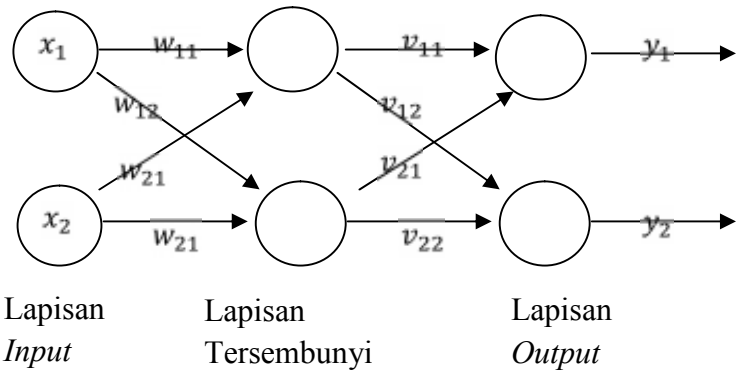


Gambar 2.11. Arsitektur *Neural Network* Lapisan Tunggal.

b. *Neural Network* dengan Banyak Lapisan (*Multilayer Net*)

Neural network dengan banyak lapisan memiliki satu atau lebih lapisan yang terletak diantara lapisan *output* dan *input*. Jaringan ini memiliki satu atau lebih lapisan tersembunyi. Umumnya terdapat lapisan bobot-bobot yang terletak antara 2 lapisan yang bersebelahan. Jaringan dengan banyak lapisan ini dapat menyelesaikan permasalahan yang lebih sulit dan teliti jika dibandingkan dengan lapisan tunggal. Tentu dengan

pembelajaran yang lebih rumit. Gambar 2.12 merupakan salah satu contoh arsitektur *neural network* dengan banyak lapisan

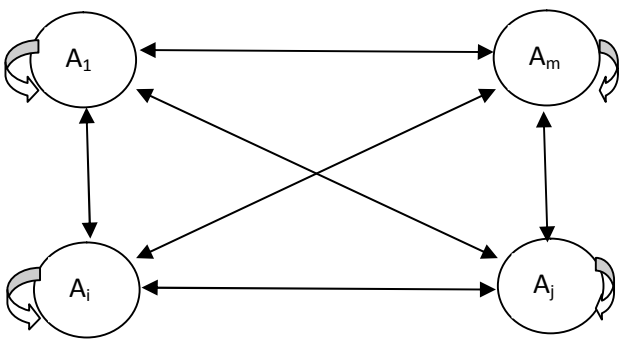


Gambar 2.12. Arsitektur *Neural Network* dengan Banyak Lapisan

Hanya ada 1 lapisan tersembunyi pada arsitektur jaringan pada gambar 2.12. Dari gambar 2.12, w_{ij} adalah bobot-bobot yang menghubungkan antara *neuron*-*neuron* pada lapisan *input* dengan lapisan tersembunyi sedangkan v_{jk} adalah bobot-bobot yang menghubungkan antara *neuron*-*neuron* pada lapisan tersembunyi dengan lapisan *output.*

c. *Neural Network* dengan Lapisan Kompetitif (*Competitive Layer Net*)

Arsitektur *neural network* dengan lapisan kompetitif memiliki bentuk yang berbeda. Antar *neuron* pada jaringan ini dapat saling dihubungkan. Pada jaringan ini sekumpulan neuron bersaing untuk menjadi aktif. Hanya ada satu neuron yang akan menghasilkan *output* yang tidak nol (Fausett, 1994:156). Gambar 2.13 merupakan salah satu contoh arsitektur *neural network* dengan lapisan kompetitif.



Gambar 2.13. Arsitektur *Neural Network* Lapisan Kompetitif.

3. Algoritma Pembelajaran

Pembelajaran merupakan bagian penting dari *neural network* karena tujuan utama dari pembelajaran adalah melakukan pengaturan terhadap bobot-bobot yang ada pada jaringan. Sehingga dari proses pembelajaran diperoleh bobot akhir yang tepat sesuai dengan pola yang dilatih. Selama proses pembelajaran, akan terjadi perbaikan bobot-bobot berdasarkan algoritma tertentu.

Menurut Fausset (1994: 12-15) terdapat 2 metode pembelajaran yaitu metode pembelajaran terawasi (*supervised learning*) dan metode pembelajaran tak terawasi (*unsupervised learning*).

a. Pembelajaran terawasi

Sebuah algoritma pembelajaran disebut terawasi jika *output* yang diharapakan telah diketahui sebelumnya. Pada algoritma ini, satu pola *input* akan diberikan ke satu neuron pada lapisan *input*. Kemudian pola tersebut akan dirambatkan ke sepanjang jaringan hingga akhirnya sampai ke *neuron* pada lapisan *output*. Lapisan *output* akan mencocokkan pola tersebut dengan pola target. *Error* akan muncul jika pada proses pencocokan ini terdapat perbedaan antara pola *output* dan pola target. Jika nilai *error* besar, maka masih perlu dilakukan pembelajaran lagi.

b. Pembelajaran tak terawasi

Berbeda dengan pembelajaran terawasi, pada pembelajaran tak terawasi tidak memerluakan target *output*. Selama proses pembelajaran, nilai bobot disusun dalam suatu *range* tertentu tergantung pada nilai *input* yang diberikan. Karena tujuan utama dari pembelajaran ini adalah mengelompokkan neuron-neuron yang hampir sama dalam suatu area tertentu, maka pembelajaran ini sangat cocok untuk pengelompokan (*klasifikasi*).

G. **Himpunan *Fuzzy***

1. Konsep Dasar Himpunan *Fuzzy*

Teori himpunan *fuzzy* adalah pengembangan dari teori himpunan klasik. Berikut akan dibahas mengenai himpunan klasik, himpunan *fuzzy* serta karakteristik himpunan *fuzzy* secara umum.

Ibrahim (2004:23) mendefinisikan himpunan klasik sebagai kumpulan dari objek yang memiliki kesamaan karakteristik. Contoh himpunan antara lain himpunan bilangan asli, himpunan mahasiswa matematika dan himpunan politisi. Objek-objek tersebut yang selanjutnya disebut anggota himpunan. Terdapat 2 nilai keanggotaan suatu anggota himpunan, yaitu menjadi anggota himpunan A ($x \in A$) atau tidak menjadi anggota himpunan A ($x \notin A$). Nilai keanggotaan dari himpunan klasik adalah {0, 1} dan didefinisikan sebagai berikut (Ibrahim, 2004:24)

$$\mu_A(x) = \begin{cases} 1, x \in A, \text{untuk semua nilai } x \\ 0, x \notin A, \text{untuk semua nilai } x \end{cases} \quad (2.33)$$

Nilai keanggotaan suatu objek adalah 1 jika objek tersebut anggota suatu himpunan dan 0 jika objek tersebut bukan anggota suatu himpunan.

Contoh 2.1 Misalkan terdapat himpunan $A = \{2,4,6,8\}$ dapat diketahui bahwa

Nilai keanggotaan 2 pada himpunan A=$\mu_A(2) = 1$ karena $2 \in A$

Nilai keanggotaan 9 pada himpunan A=$\mu_A(9) = 0$ karena $9 \notin A$

Tahun 1965 Lotfi A. Zadeh memperkenalkan teori himpunan *fuzzy* sebagai cara untuk merepresentasikan kesamaran dalam bahasa (Lin & Lee, 1996:3). Himpunan *fuzzy* merupakan pengembangan dari teori himpunan klasik. Pada Teori himpunan klasik, keanggotaan suatu objek hanya berkisar antara anggota himpunan dan bukan anggota himpunan. Sedangkan pada himpunan *fuzzy*, nilai keanggotaan dari suatu himpunan adalah berkisar antara 0 sampai 1 (*range* [0 1]).

Menurut Zimmermann (1991:11-12) jika *X* adalah koleksi dari objek-objek yang dinotasikan secara generik oleh *x*, maka himpunan *fuzzy* $\tilde{A}$, dalam *X* adalah suatu himpunan pasangan berurutan:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \square X\} \quad (2.34)$$

dengan $\mu_{\tilde{A}}(x)$ adalah derajat keanggotaan x di $\tilde{A}$ yang memetakkan X ke ruang keanggotaan M yang terletak pada rentang [0, 1].

Contoh 2.2

Misalkan T menyatakan suhu dalam Celcius. Anggota T adalah

$$T = \{0, 5, 10, 15, 20, 25, 30, 35, 40\}$$

Tingkat panas suhu dapat di definisikan dengan himpunan *fuzzy* sebagai berikut

$p =$

$\{(0,0), (5,0.1), (10,0.3), (15,0.5), (20,0.6), (25,0.7), (30,0.8), (35, 0.9), (40,1.0)\}$

.

Himpunan *fuzzy* merepresentasikan bahwa suhu 0^oC tidak panas. Suhu $5\ ^oC$, $10\ ^oC$ dan $15\ ^oC$ agak panas dan $40\ ^oC$ panas.

Himpunan *fuzzy* dapat dituliskan sebagai pasangan berurutan. Elemen pertama menunjukkan nama elemen dan elemen kedua menunjukkan nilai keanggotaannya.

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \square X\} \quad (2.35)$$

Contoh 2.2 menunjukkan notasi pasangan berurutan himpunan *fuzzy*.

Selain itu, himpunana *fuzzy* dinotasikan dengan (Lin & Lee, 1996:13)

$$\tilde{A} = \mu_1/x_1 + \mu_1/x_1 + \mu_2/x_3 + \cdots = \sum_{i=1}^{n} \mu_i/x_i \text{ untuk } X \text{ diskret} \quad (2.36)$$

$$\tilde{A} = \mu_1/x_1 + \mu_1/x_1 + \mu_2/x_3 + \cdots = \int_x \mu_i/x_i \text{ untuk } X \text{ kontinu} \quad (2.37)$$

Simbol " / " bukan merupakan operasi pembagian, begitu juga simbol "+" bukan merupakan operasi penjumlahan.

Contoh.2.3 Misalkan X = {25; 33; 67; 46; 27; 55} merupakan himpunan semesta dari umur. Didefinisikan himpunan *fuzzy* umur muda, umur paruh baya dan umur tua dengan ketentuan

$$\mu_{muda}(x) = \begin{cases} 1; & x \leq 25 \\ \frac{45-x}{20}; & 25 \leq x \leq 45 \\ 0; & x \geq 45 \end{cases}$$

$$\mu_{parobaya}(x) = \begin{cases} 0; & x \leq 35 \; atau \; x \geq 55 \\ \frac{x-35}{20}; & 35 \leq x \leq 45 \\ \frac{55-x}{10}; & 45 \leq x \leq 55 \end{cases}$$

$$\mu_{tua}(x) = \begin{cases} 0; & x \leq 45 \\ \frac{x-45}{20}; & 45 \leq x \leq 65 \\ 1; & x \geq 65 \end{cases}$$

Dari ketentuan tersebut dapat diketahui bahwa

Tabel 2.1 Fungsi Keanggotaan *Fuzzy* Muda, Paruh Baya, dan Tua

Umur	Umur muda$(\mu_{\tilde{A}}(X))$	Umur paruh baya $(\mu_{\tilde{B}}(X))$	Umur tua $(\mu_{\tilde{C}}(X))$
25	1	0	0
33	0,6	0	0
67	0	0	1
46	0	0,9	0,05
27	0,9	0	0
55	0	0	0,5

Himpunan *fuzzy* untuk muda dapat dinotasikan dengan

$$\tilde{A} = 1/25 + 0{,}6/33 + 0/67 + 0/46 + 0{,}9/27 + 0/55$$

Seperti pada himpunan klasik, pada himpunan *fuzzy* sifat-sifat tersebut dinyatakan dalam definisi berikut (Ibrahim, 2004:35-36)

Definisi 2.1

Himpunan *fuzzy* dikatakan kosong jika dan hanya jika nilai dari fungsi keanggotaan untuk semua anggota adalah 0.

Definisi 2.2

Himpunan *fuzzy* dikatakan sebagai himpunan semesta *fuzzy* jika dan hanya jika nilai fungsi keanggotaan dari semua anggota adalah 1.

Definisi 2.3

Dua himpunan *fuzzy A* dan *B* dikatakan sama jika $\mu_A(x) = \mu_B(x)$ untuk semua $x \in X$

Definisi 2.4

Himpunan α-level adalah himpunan elemen-elemen yang ada pada himpunan *fuzzy* $\tilde{A}$, sedemikian sehingga untuk suatu nilai α

$$A_\alpha = \{x \in X | \mu_{\tilde{A}}(x) \geq \alpha\} \quad (2.38)$$

Contoh 2.4 misalkan $\tilde{A} = 1/25 + 0{,}6/33 + 0{,}1/67 + 0{,}2/46 + 0{,}9/27 + 0{,}3/55$ dapat disimpulkan

untuk nilai $\alpha = 0{,}1$; maka $A_{0,1} = \{25, 33, 67, 46, 27, 55\}$

untuk nilai $\alpha = 0{,}2$; maka $A_{0,2} = \{25, 33, 46, 27, 55\}$

untuk nilai $\alpha = 0{,}3$; maka $A_{0,3} = \{25, 33, 27, 55\}$

untuk nilai $\alpha = 0{,}6$; maka $A_{0,6} = \{25, 33, 27\}$

untuk nilai $\alpha = 0{,}9$; maka $A_{0,9} = \{25, 27\}$

untuk nilai $\alpha = 1{,}0$; maka $A_{1,0} = \{25\}$

Definisi 2.5

Support dari himpunan *fuzzy* $\tilde{A}$, $Supp(\tilde{A})$ adalah himpunan crisp dari $x \in X$ sedemikian hingga $\mu_{\tilde{A}}(x) > 0$ (Ibrahim, 2004:36).

Contoh 2.5

Dari contoh 2.4, {25, 33, 27} adalah *support* dari himpunan *fuzzy* muda $\tilde{A}$. Sedangkan {55, 46, 67} bukan merupakan *support* dari himpunan *fuzzy* muda $\tilde{A}$.

Definisi 2.6

Core dari himpunan *fuzzy A*, *core*(*A*) adalah himpunan klasik dari semua $x \in X$ sehingga $\mu_A(x) = 1$. *Core* dari himpunan *fuzzy* mungkin merupakan himpunan kosong.

Definisi 2.7

Height, *h(A)* dari himpunan *fuzzy A* adalah nilai tertinggi dari μ_A yang nilai α-level-nya tidak kosong.

2. Fungsi Keanggotaan

Fungsi keanggotaan (*membership function*) adalah suatu kurva yang menunjukkan pemetaan dari titik-titik *input* data terhadap nilai keanggotaannya. Ada beberapa fungsi yang dapat digunakan untuk mendapatkan nilai keanggotaan dari titik-titik *input*. Fungsi-fungsi tersebut antara lain (Kusumadewi dan Hartati, 2010:22-37): Representasi Linear, Representasi Kurva Segitiga, Representasi Kurva Trapesium, Representasi Kurva Bentuk Bahu, Representasi kurva-S, Representasi Kurva Bentuk Lonceng (*Bell Curve*).

a. Representasi Linear

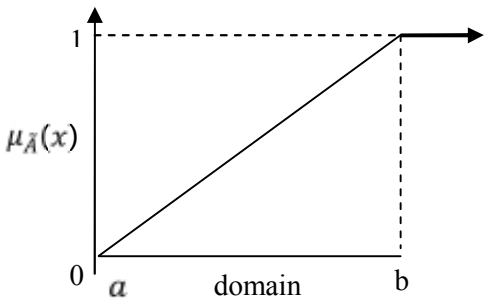
Bentuk yang paling sederhana dari fungsi keanggotaan adalah representasi linear. Dalam reprtesentasi linear, pemetaan *input* terhadap nilai keanggotaannya digambarkan sebagai garis lurus. Terdapat 2 kondisi dari *fuzzy* yang linear.

1) Kondisi pertama adalah kenaikan *input* yang memiliki derajat keanggotaan [0] bergerak naik menuju nilai domain yang memiliki derajat keanggotaan lebih tinggi. Gambar 2.14 Menunjukkan keadaan ini. Fungsi keanggotaan untuk linear naik adalah:

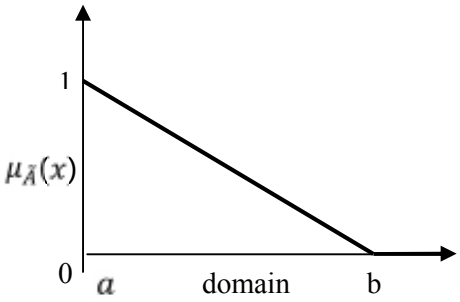
$$\mu_{\tilde{A}}(x) = \begin{cases} 0; & x \leq a \\ \frac{x-a}{b-a}; & a < x < b \\ 1; & x \geq b \end{cases} \tag{2.39}$$

2) Kondisi kedua adalah kebalikan kondisi yang pertama. Garis lurus dimulai dari domain yang memiliki nilai keanggotaan [1] yang kemudian turun menuju domain yang memiliki nilai keanggotaan lebih rendah. Gambar 2.15 menunjukkan kondisi ini. Fungsi keanggotaan linear turun adalah

$$\mu_{\tilde{A}}(x) = \begin{cases} \frac{b-x}{b-a}; & a \le x \le b \\ 0; & x \ge b \end{cases} \quad (2.40)$$



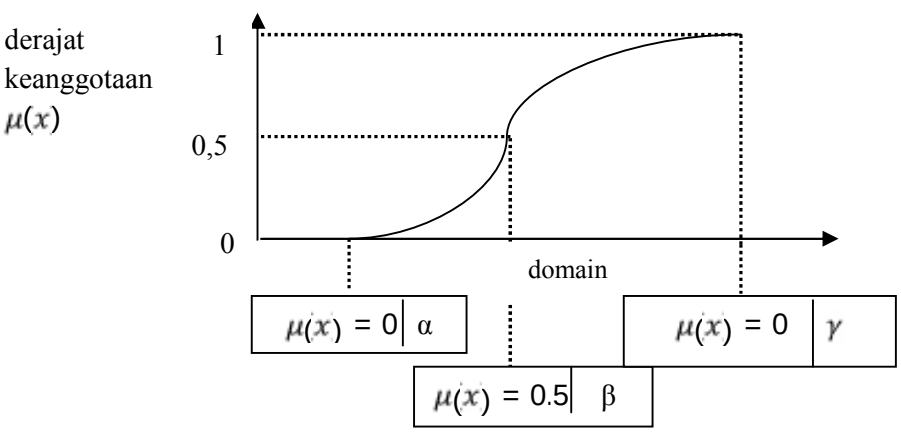
Gambar 2.14 Representasi Linear Naik



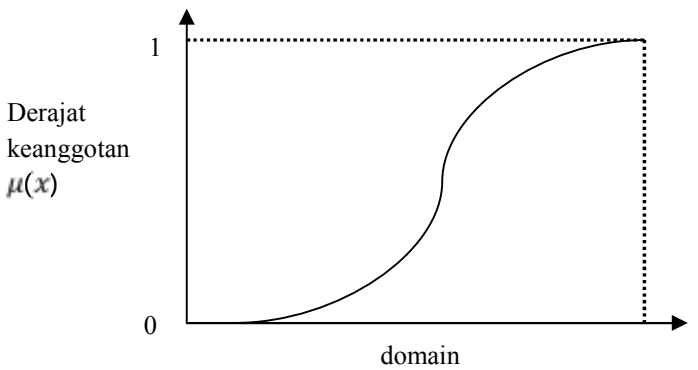
Gambar 2.15 Representasi Linear Turun

b. Representasi Kurva-S

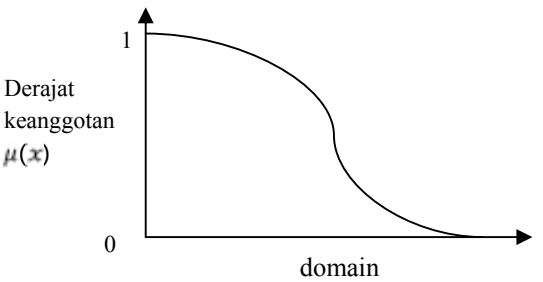
Kurva-S atau *sigmoid* menggambarkan pertumbuhan dan penyusutan yang tidak linear. Kurva-S pertumbuhan akan bergerak dari domain yang memiliki nilai keanggotaan 0 menuju domain dengan nilai keanggotaan 1. Sedangkan kurva penyusutan bergerak dari domain yang memiliki nilai keanggotaan 1 menuju domain yang memiliki nilai keanggotaan 0. Titik *infleksi* adalah titik dimana fungsi keanggotaan bertumpu pada 50% nilai keanggotaannya. Kurva-S didefinisikan dengan menggunakan 3 parameter, yaitu: nilai keanggotaan nol (α), nilai keanggotaan lengkap (γ), dan titik infleksi (β). Gambar 2.19 menunjukkan karakteristik kurva-S. Gambar 2.20 menunjukkan kurva-S pertumbuhan. Gambar 2.21 menunjukkan kurva-S penyusutan.



Gambar 2.16. Karateristik Fungsi Kurva-S



Gambar 2.17 Kurva-S Pertumbuhan



Gambar 2.18 Kurva-S Penyusutan

Fungsi keanggotaan pada kurva Pertumbuhan:

$$S(x;\alpha,\beta,\gamma)=\begin{cases}0; & x\le\alpha\\ 2((x-\alpha)/(\gamma-\alpha))^2; & \alpha<x\le\beta\\ 1-2((\gamma-x)/(\gamma-\alpha))^2; & \beta<x<\gamma\\ 1; & x\ge\gamma\end{cases} \tag{2.41}$$

Fungsi keanggotaan kurva penyusutan adalah

$$S(x;\alpha,\beta,\gamma)=\begin{cases}1; & x\le\alpha\\ 1-2((x-\alpha)/(\gamma-\alpha))^2; & \alpha<x<\beta\\ 2((\gamma-x)/(\gamma-\alpha))^2; & \beta\le x<\gamma\\ 0; & x\ge\gamma\end{cases}. \tag{2.42}$$

3. Operator-Operator Himpunan *Fuzzy*

Himpunan *fuzzy* adalah pengembangan dari himpunan klasik. Beberapa sifat dan karakteristik dari himpunan klasik juga dimiliki oleh himpunan *fuzzy*. Dalam himpunan klasik, dikenal beberapa operator antara lain komplemen, *union* dan interseksi. Dalam himpunan *fuzzy*, juga dikenal beberapa operator. Salah satunya adalah operator-operator dasar Zadeh (Klir, 1997:90-93). Operator-operator dasar Zadeh adalah operator yang ditemukan oleh Zadeh. Nilai keanggotaan sebagai hasil dari 2 operasi dikenal dengan nama *fire strength* atau α-predikat. Terdapat 3 operator dasar Zadeh antara lain *AND, OR,* dan *NOT* (Kusumadewi & Hartati, 2010:38).

a. Operator *AND*

Operator *AND* sama halnya operator interseksi dalam himpunan klasik. Nilai *fire strength* pada operasi ini diperoleh dengan mengambil nilai terkecil dari nilai keanggotaan antar elemen pada himpunan-himpunan yang bersangkutan.

$$\mu_{\tilde{A}\cap\tilde{B}}(x,y) = \min\left(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y)\right) \quad (2.43)$$

b. Operator *OR*

Operator *OR* berhubungan dengan operator *union* pada himpunan klasik. Jika pada operator *AND*, *fire strength* diperoleh dengan mengambil nilai minimum dari nilai keanggotaan antar elemen, pada operator *OR* nilai *fire strength* diperoleh dari nilai maksimum dari nilai keanggotaan antar elemen yang bersangkutan.

$$\mu_{\tilde{A}\cup\tilde{B}}(x,y) = \max\left(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y)\right) \quad (2.44)$$

c. Operator *NOT*

Operator *NOT* berhubungan dengan operator komplemen pada himpunan klasik. *Fire strength* diperoleh dengan mengurangkan nilai keanggotaan elemen yang bersangkutan terhadap 1.

$$\mu_{\tilde{A}'}(x) = 1 - \mu_{\tilde{A}}(x) \quad (2.45)$$

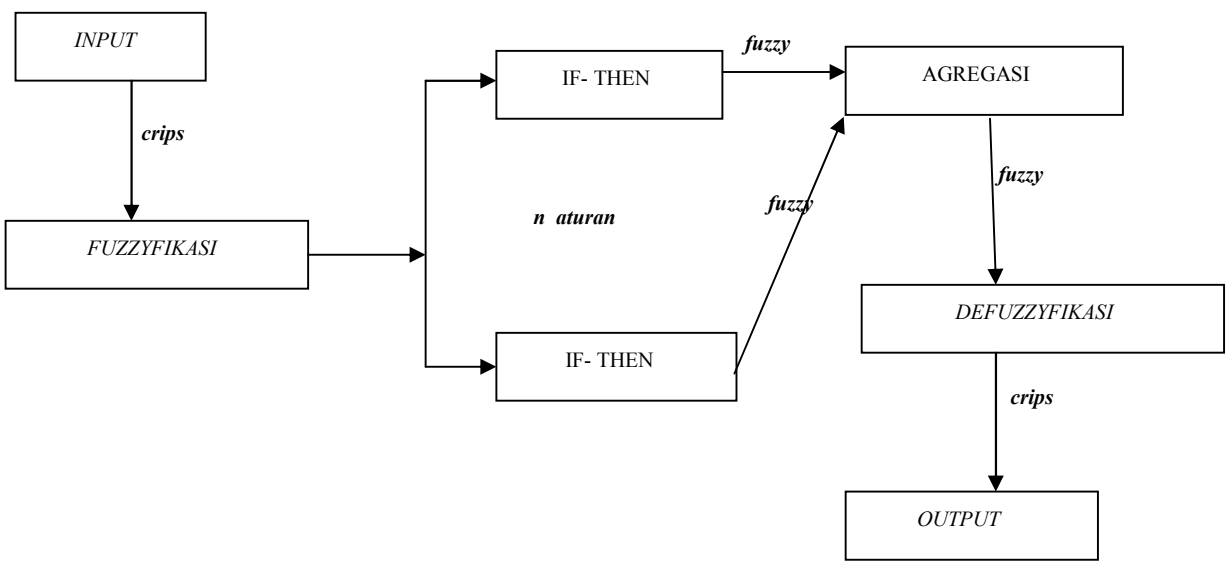
Contoh 2.6 Misalkan diketahui $\mu_{\tilde{A}}(muda)$=0,4 . Maka nilai keanggotaan untuk tidak muda adalah

$$\mu_{\tilde{A}'}(muda) = 1 - \mu_{\tilde{A}}(muda) = 1 - 0{,}4 = 0{,}6$$

4. Sistem Inferensi *Fuzzy*

Sistem Inferensi *Fuzzy* (*Fuzzy Inference System* /*FIS*) disebut juga *fuzzy inference engine* adalah sistem yang dapat melakukan penalaran dengan prinsip serupa seperti manusia melakukan penalaran dengan nalurinya. Menurut Kusumadewi dan Hartati (2010: 40) Sistem inferensi *fuzzy* adalah suatu kerangka komputasi yang didasarkan pada teori himpunan *fuzzy*, aturan *fuzzy* berbentuk *IF- THEN*, dan penalaran *fuzzy*.

Terdapat beberapa jenis *FIS* yang dikenal yaitu Mamdani, Sugeno dan Tsukamoto. *FIS* yang paling mudah dimengerti adalah *FIS* Mamdani. Proses dalam *FIS* ditunjukan pada Gambar 2.19. *Input* yang diberikan kepada *FIS* adalah berupa bilangan tertentu dan *output* yang dihasilkan juga harus berupa bilangan tertentu. *FIS* menerima *input* klasik, *input* ini kemudian dikirim ke basis pengetahuan yang berisi *n* aturan *fuzzy*. Aturan-aturan tersebut berbentuk *IF- THEN*. Setiap aturan akan menghasilkan *fire strength*. Apabila jumlah aturan yang tersedia lebih dari satu, maka akan dilakukan agregasi atau penggabungan dari semua aturan. Selanjutnya pada hasil agregasi akan dilakukan *defuzzy* untuk mendapatkan nilai klasik sebagai *output* sistem.



Gambar 2.19 Diagram Blok *FIS*

Dalam tugas akhir ini akan digunakan metode Sugeno (TSK) sebagai proses inferensi. Sistem inferensi *fuzzy* metode Takagi-Sugeno-Kang (TSK) merupakan metode inferensi *fuzzy* untuk aturan yang direpresentasikan dalam bentuk *IF– THEN*, dimana *output* (konsekuen) sistem tidak berupa himpunan *fuzzy*, melainkan berupa konstanta atau persamaan linear. Variabel dari persamaan linear tersebut sesuai dengan variabel-variabel *inputnya*. Model ini diperkenalkan oleh Takagi-Sugeno Kang pada tahun 1985. Ada 2 model pada metode TSK, yaitu (Kusumadewi & Hartati, 2010:53)

a. Metode *Fuzzy* Sugeno Orde-0

   Secara umum bentuk model *fuzzy* Sugeno Orde-0 adalah

   $$IF\ (x_1 is\ A_1) \circ (x_2 is\ A_2) \circ \dots \circ (x_i is\ A_i)\ THEN\ y = k \quad (2.46)$$

   dengan A_i adalah himpunan *fuzzy* ke-*i* sebagai antesenden, x_j adalah variabel *input* ke-*j*, *k* adalah konstanta klasik sebagai konsekuen, dan $\circ$ adalah operator *fuzzy* (seperti *AND* atau *OR*).

b. Model *Fuzzy* Sugeno Orde-1

   Secara umum bentuk model *fuzzy* sugeno orde-1 adalah

   $$IF\ (x_1 is\ A_1) \circ (x_2 is\ A_2) \circ \dots \circ (x_i is\ A_i)\ THEN\ y = p_1 \square x_1 + \cdots + p_i \square x_i + q \quad (2.47)$$

   x_i adalah input data ke-*i*, A_i adalah himpunan *fuzzy* ke-*i* sebagai antesenden, $\circ$ adalah operator *fuzzy* (seperti AND atau OR), p_i adalah suatu konstanta klasik ke-*i* dan *q* juga merupakan konstanta untuk persamaan linear dalam konsekuen.

   Apabila fire *strength*(α_r) dan nilai y_r untuk setiap aturan ke-*r* telah diperoleh (r = 1, ..., R), selanjutnya akan dilakukan proses agregasi aturan. Proses agregasi dilakukan dengan cara melakukan penjumlahan hasil perkalian antara *fire strength* dengan nilai *y* tersebut. Proses penegasan (*defuzzy*) dilakukan dengan menggunakan konsep rata-rata tertimbang

(weighted average), seperti terlihat pada persamaan (Kusumadewi & Hartati, 2010:54)

$$y_i^* = \frac{\sum_{i=1}^{n} \alpha_i y_i}{\sum_{i=1}^{n} \alpha_i} \quad (2.48)$$

dengan:

y^* = nilai o*utput*

α_i = nilai *fire strength* ke-i

y_i = nilai *output* aturan ke-i

n = banyaknya aturan *fuzzy IF–THEN*

Contoh 2.7 Misalkan terdapat ketentuan untuk tiga nilai yaitu Aljabar, Geometri, Statistik sebagai berikut

$$\mu_{rendah}(x) = \begin{cases} 1; & x \leq 35 \\ \frac{55-x}{20}; & 35 \leq x \leq 55 \\ 0; & x \geq 55 \end{cases}$$

$$\mu_{cukup}(x) = \begin{cases} 0; & x \leq 55 \; atau \; x \geq 75 \\ \frac{x-55}{20}; & 55 \leq x \leq 65 \\ \frac{75-x}{10}; & 65 \leq x \leq 75 \end{cases}$$

$$\mu_{tinggi}(x) = \begin{cases} 0; & x \leq 75 \\ \frac{x-75}{20}; & 75 \leq x \leq 90 \\ 1; & x \geq 90 \end{cases}$$

Terdapat 3 aturan sebagai berikut, (A: aljabar, G:geometri, S: statistic, T; tinggi, C: cukup, dan R: rendah)

$Aturan 1$:

R_1: $IF\ A\ is\ T\ AND\ G\ is\ T\ AND\ S\ is\ T\ THEN\ y_1\ is$ 100

$Aturan 2$:

R_2: $IF\ A\ is\ C\ AND\ G\ is\ C\ AND\ S\ is\ C\ THEN\ y_1\ is$ $0{,}33A + 0{,}33G + 0{,}33S - 0{,}06$

$Aturan 3$:

R_3: $IF\ A\ is\ R\ AND\ G\ is\ R\ AND\ S\ is\ R\ THEN\ y_1\ is$ $0{,}25A + 0{,}25G + 0{,}25S - 0{,}03$

Misalkan seorang mahasiswa memperoleh nilai Aljabar 77, Geometri 98, dan Statistik 88. Derajat keanggotaan dari setiap himpunan adalah

$\mu_{aljabar}(rendah) = 0$; $\mu_{geometri}(rendah) = 0$; dan

$\mu_{statistik}(rendah) = 0$

$\mu_{aljabar}(cukup) = 0$; $\mu_{geometri}(cukup) = 0$; dan $\mu_{statistik}(cukup) = 0$

$\mu_{aljabar}(tinggi) = 0{,}1$; $\mu_{geometri}(tinggi) = 1$; dan

$\mu_{statistik}(tinggi) = 0{,}65$

Nilai untuk masing-masing y_i pada tiap aturan *fuzzy* adalah sebagai berikut:

Aturan 1:

R_1: $IF\ A\ is\ T\ AND\ G\ is\ T\ AND\ S\ is\ T\ THEN\ y_1\ is\ 100$

Sehingga diperoleh

Nilai *fire strenght* adalah $\alpha_1 = min(\mu_{R_A}; \mu_{R_G}; \mu_{R_S}) = min(0; 0; 0) = 0$

Nilai y_1 adalah $y_1 = 100$

Aturan 2:

R_2: $IF\ A\ is\ C\ AND\ G\ is\ C\ AND\ S\ is\ C\ THEN\ y_1\ is\ 0{,}33A + 0{,}33G + 0{,}33S - 0{,}06$

Sehingga diperoleh

Nilai *fire strenght* adalah $\alpha_2 = min(\mu_{C_A}; \mu_{C_G}; \mu_{C_S}) = min(0; 0; 0) = 0$

Nilai y_2 adalah

$y_2 = 0{,}33A + 0{,}33G + 0{,}33S - 0{,}06 = 0{,}33 \times (77 + 98 + 88) - 0{,}06$

$= 86{,}73$

Aturan 3:

R_3: $IF\ A\ is\ R\ AND\ G\ is\ R\ AND\ S\ is\ R\ THEN\ y_1\ is\ 0{,}25A + 0{,}25G + 0{,}25S - 0{,}03$

Sehingga diperoleh

Nilai *fire strength* $\alpha_3 = min(\mu_{T_A}; \mu_{T_G}; \mu_{T_S}) = min(0{,}1; 1; 0{,}65) = 0{,}1$

Nilai y_3 adalah

$y_3 = 0{,}25A + 0{,}25G + 0{,}25S - 0{,}06 = 0{,}25 \times (77 + 98 + 88) - 0{,}03$

$= 65{,}72$

Nilai y^* dengan rata-rata terbobot adalah

$y^* = \frac{(0)(100)+(0)(86{,}73)+(0{,}1)(65{,}72)}{0+0+0{,}1} = 65{,}72$

dengan aturan tersebut, nilai akhir mahasiswa adalah 65,72

H. *Neuro Fuzzy*

Logika *fuzzy* dan *neural network* adalah komponen utama pembentuk *soft computing*. Keduanya mempunyai ciri khas serta cara yang berbeda untuk melakukan sebuah perhitungan. *Neural network* adalah model yang memiliki struktur yang sangat baik karena meniru jaringan pada manusia. Kekurangan dari *neural network* adalah tingkat perhitungannya sendiri yang tergolong rendah (Lin & Lee, 1996:481). Logika *fuzzy* memiliki tingkat penalaran yang lebih tinggi dibandingkan *neural network*. Latar belakang inilah yang menyebabkan terciptanya suatu sistem yang menggabungkan *neural network* dan logika *fuzzy* yaitu *neural fuzzy* system(NFS).

Sistem *neuro fuzzy* adalah sistem *fuzzy* di mana bobot-bobot yang terhubung pada jaringan tersebut dihitung dengan penalaran *fuzzy* namun pelatihan fungsi-fungsi keanggotaan dilakukan dengan algorima *backpropagation* pada *neural network*. Estimasi parameter sistem menggunakan *neural network* dan pemodelan utama menggunakan logika *fuzzy*. *Neural network* pada *neural fuzzy sistem* digunakan sebagai pengendali penalaran *fuzzy*. Konsep dasar penggunaan *neural network* sebagai pengendali penalaran *fuzzy* adalah menggunakan *neural network* untuk merealisasikan atau membangkitkan sistem inferensi *fuzzy* model sugeno. Pada bagian antesenden, penggunaan *neural network* digunakan untuk membangkitkan

fungsi keanggotaan. Sedangkan pada bagian konsekuen, *neural network* digunakan untuk melakukan inferensi. Hal tersebut dilakukan karena salah satu kelemahan sistem inferensi *fuzzy* adalah penentuan fungsi keanggotaan dan pembangkitan fungsi pembelajaran pada aturan-aturan inferensi.

Sistem *neuro fuzzy* model Soegeno yang dikembangkan oleh Takagi Sugeno dan Hayashi pada tahun 1991 menggunakan *neural network* dengan algoritma pembelajaran *backpropagation* untuk membangun himpunan-himpunan *fuzzy* pada bagian antesenden dan fungsi inferensi yang ada pada bagian konsekuen. Aturan yang dipakai adalah sebagai berikut (Lin & Lee, 1996:507):

$$R^s: IF\ x = (x_1, x_2, \dots, x_n)\ is\ A_s\ THEN\ y_s = NN_s(x_1, x_2, \dots, x_n) \quad (2.49)$$

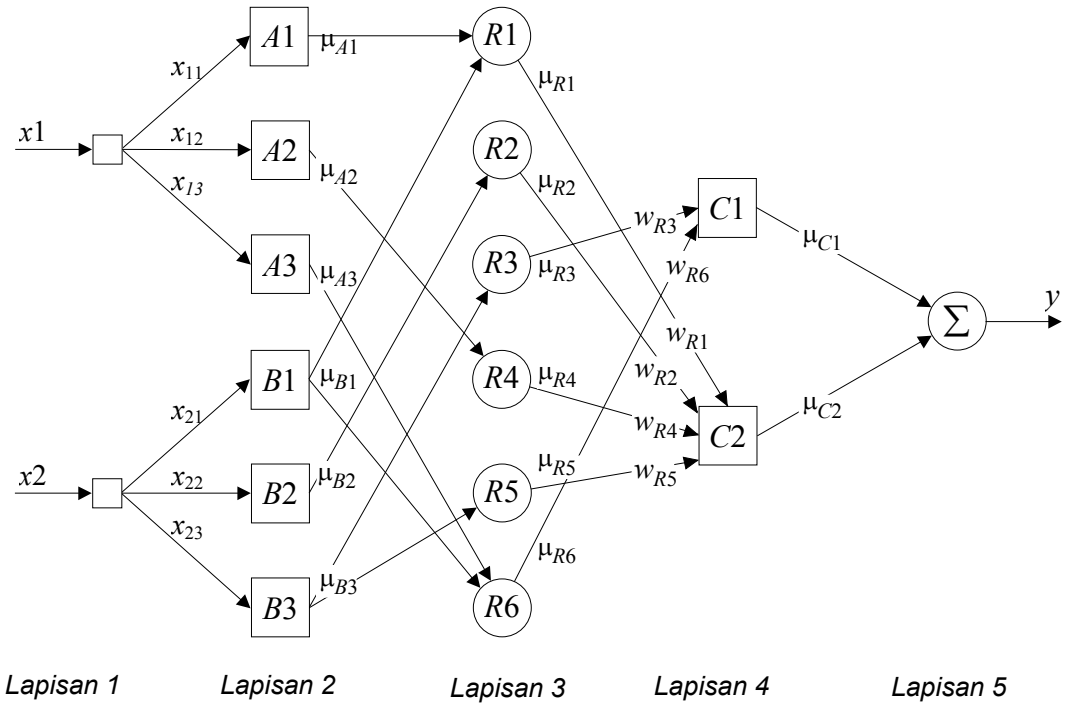
dengan

s : aturan inferensi *fuzzy* ke-s

A_s : himpunan *fuzzy* pada aturan ke-s bagian anteseden

NN_s : jaringan *backpropagation* dengan *input* ($x_1, x_2, ..., x_m$) dan *output* y_s.

Secara garis besar, cara kerja sistem *neuro fuzzy* secara umum di tunjukkan pada Gambar 2. 26 (Lin & Lee, 1996: 536).



Gambar 2.20. Cara Kerja Sistem *Neuro Fuzzy* Secara Umum

pada Gambar 2.20

x : *input* jaringan

μ : nilai keanggotaan himpunan *fuzzy*

y : *output*

Setiap lapisan pada sistem *neuro fuzzy* dikaitkan dengan langkah-langkah tertentu dalam proses inferensinya. Dari Gambar 2.26 (Lin & Lee, 1996: 536-537)

1. Lapisan 1 adalah lapisan *input* (lapisan linguistik). Setiap *neuron* dalam lapisan ini mengirimkan sinyal langsung ke lapisan berikutnya .
2. Lapisan 2 adalah lapisan *fuzzifikasi*. *Neuron* pada lapisan ini merupakan representasi dari himpunan *fuzzy* yang digunakan sebagai anteseden dalam

aturan *fuzzy*. *Neuron* pada lapisan *fuzzifikasi* menerima *input* berupa crips dan menentukan nilai keanggotaan setiap *input*.

3. Lapisan 3 adalah lapisan dengan aturan *fuzzy*. Setiap neuron dalam lapisan ini mempunyai aturan-aturan *fuzzy*. Aturan *neuron fuzzy* menerima *input* yang sesuai dari lapisan 2. *Input* tersebut merupakan himpunan *fuzzy* hasil *fuzzifikasi* dengan antesenden pada lapisan 2. Pada lapisan ini, *input* dimasukkan kedalam neuron-neuron yang bersesuaian yang selanjutnya dalam tugas akhir ini disebut *custer*. Misalnya pada Gambar 2.20, neuron *R1* yang bersesuaian dengan aturan 1, menerima *input* dari neuron *A1* dan *B1*. Pengklusteran menggunakan teknik yang disebut *fuzzy C-means* (FCM). FCM adalah suatu teknik pengklasteran data berdasarkan nilai keanggotaan data tersebut. Algoritma pengklasteran menggunakan FCM adalah sebagai berikut (Zimmerman, 1991:234):
   a. Menentukan matriks *X* yang berkuran $n \times m$ dengan *n* adalah jumlah data yang akan dikluster dan *m* adalah jumlah variabel (kriteria)
   b. Menentukan jumlah *cluster* yang dibentuk. Tidak ada aturan tertentu dalam penentuan jumlah *cluster*, artinya jumlah *cluster* ditentukan secara bebas asalkan *cluster* yang dibentuk lebih dari 1.
   c. Menentukan pangkat (pembobot) yang dinotasikan dengan *w*. Jumlah pembobot harus lebih dari 1.
   d. Menentukan maksimum iterasi.
   e. Menentukan kriteria penghentian yang dinotasikan dengan ξ yaitu nilai positif yang sangat kecil.
   f. Menentukan iterasi awal, $t = 1\ dan \Delta = 1$
   g. Membentuk matriks partisi awal yang dinotasikan dengan U^0 yaitu matriks dengan entri-entri yaitu $\mu_{nc}(x_c)$ = nilai keanggotaan setiap data ke-*n* pada *cluster* ke *c* dengan ketentuan sebagai berikut:

$$U^0 = \begin{bmatrix} \mu_{11}(x_1) & \mu_{12}(x_1) \cdots & \mu_{1c}(x_1) \\ \mu_{21}(x_1) & \mu_{22}(x_1) \cdots & \mu_{2c}(x_1) \\ \vdots & & \vdots \\ \mu_{n1}(x_1) & \mu_{n2}(x_1) \cdots & \mu_{nc}(x_1) \end{bmatrix} \quad (2.50)$$

h. Menghitung pusat *cluster* yang dinotasikan dengan *V*. Setiap *cluster* memiliki pusat *cluster* yang berbeda. Matriks pusat *cluster* adalah sebagai berikut

$$V = \begin{bmatrix} v_{11}^{*} & \cdots & v_{1m}^{*} \\ \vdots & & \vdots \\ v_{c1}^{*} & \cdots & v_{cm}^{*} \end{bmatrix} \quad (2.51)$$

.

$$v_{ij}^{*} = \frac{\sum_{k=1}^{n}(\mu_{ik})^{w} x_{kj}}{\sum_{k=1}^{n}(\mu_{ik})^{w}} \quad (2.52)$$

dengan:

v_{ij}^{*} = pusat *cluster* variabel ke-j pada *cluste r*ke-i

μ_{ik} = nilai keanggotaan data ke-k pada *cluster*ke-i

w = pangkat (pembobot)

x_{kj} = pengamatan ke-k pada variabel ke-j

i. Memperbaiki derajat keanggotaan setiap data pada setiap *cluster* yaitu dengan cara memperbaiki matriks partisi. Cara memperbaiki derajat keanggotaan adalah sebagai berikut:

$$\mu_{ik} = \left[\sum_{j=1}^{C} \left(\frac{d_{ik}}{d_{jk}} \right)^{2/(w-1)} \right]^{-1} . \quad (2.53)$$

dengan:

$$\boldsymbol{d_{ik} = d(x_k - v_i) = \left[\sum_{j=1}^{m}(x_{kj} - v_{ij}^{*})\right]^{1/2}} \quad (2.54)$$

dimana,

μ_{ik} = Nilai keanggotaan data ke-k pada *cluster* ke-i.

d_{ik} = Jarak antara pusat *cluster*ke-i dengan data ke-k.

j. Langkah terakhir adalah menentukan kriteria berhenti, yaitu perubahan matriks partisi pada iterasi sekarang dengan iterasi sebelumnya, sebagai berikut

$$\Delta = \| U^{t} - U^{t-1} \| \quad (2.55)$$

Apabila $\Delta \leq \xi$, maka iterasi dihentikan, namun apabila $\Delta > \xi$ maka naikkan iterasi ($t = t+1$) dan kembali ke langkah c. Pencarian nilai Δ dapat dilakukan dengan mengambil elemen terbesar dari nilai mutlak selisih antara $\mu_{ik}(t)$ dengan $\mu_{ik}(t-1)$.

4. Lapisan 4 adalah lapisan nilai keanggotaan *output*. *Neuron* pada lapisan ini adalah himpunan *fuzzy* yang digunakan sebagai konsekuen pada aturan *fuzzy*.
5. Lapisan 5 adalah lapisan *defuzzyfikasi*. Pada lapisan ini diperoleh *output* dari sistem *neuro fuzzy*.

Pemodelan *fuzzy* didasarkan pada sistem yang hendak dibangun menggunakan sistem inferensi *fuzzy* (Lin, 1996:511). Mengidentifikasi aturan-aturan *fuzzy* dan membangkitkan fungsi keanggotaan bukanlah hal yang mudah. Horikawa pada tahun 1992 memperkenalkan metode pemodelan *fuzzy* melalui pembelajaran *neural network* algoritma *backpropagation*. Terdapat 3 tipe *fuzzy* modeling *network* (FMN) yang diperkenalkan Horikawa yaitu FMN tipe I, FMN tipe II, FMN tipe III. Pada tugas akhir ini digunakan FMN tipe III.

FMN tipe III yaitu aturan *fuzzy* yang bentuk konsekuennya berupa persamaan linear orde pertama. Format aturan untuk FMN tipe III adalah sebagai berikut

$$R^{i}: IF\ x_1\ is\ A_{i1}\ AND\ x_2\ is\ A_{i2}\ THEN\ y = f_i(x_1, x_2);\ i = 1,2,\dots,n \quad (2.56)$$

Output jaringan dihitung dengan rumus:

$$y^* = \frac{\sum_{i=1}^{r} \mu_i f_i(x_1, x_2)}{\sum_{i=1}^{r} \mu_i} = \sum_{i=1}^{r} \hat{\mu}_i f_i(x_1, x_2) \quad (2.57)$$

$f_i(x_1, x_2) = a_{i0} + a_{i1}x_1 + a_{i2}x_2$; a_{ij}, $j = 0,1,2$ adalah konstanta.

R^i : aturan *fuzzy* ke-i

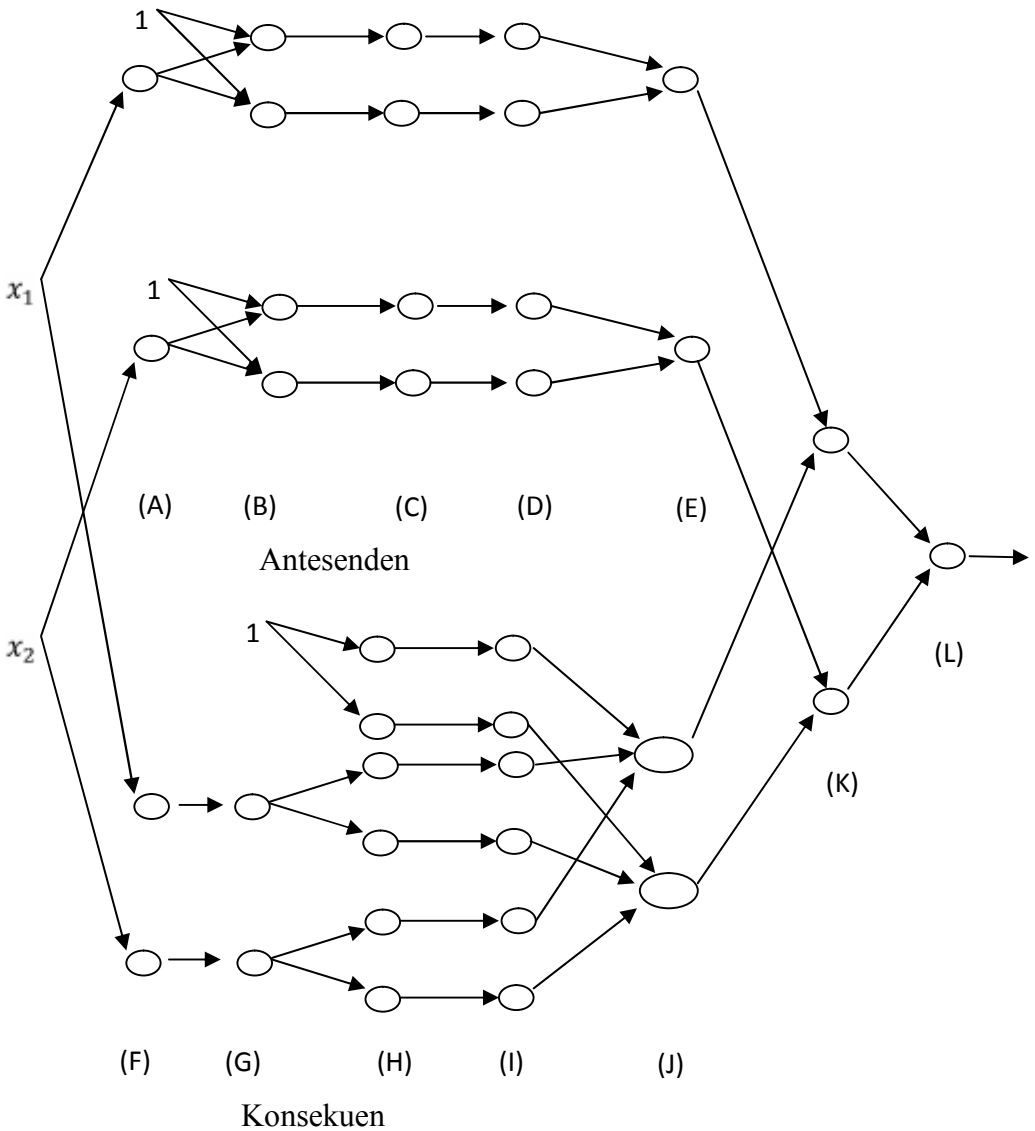
A_{i1} : himpunan *fuzzy* ke-i pada bagian anteseden untuk x_1

A_{i2} : himpunan *fuzzy* ke-i pada bagian anteseden untuk x_2

$f_i(x_1,x_2)$: persamaan linear orde satu untuk aturan ke-i

y^* : *output* jaringan lapisan.

Gambar 2.21 adalah arsitektur jaringan *neuro fuzzy* menggunakan FMN III (Lin & Lee, 1996: 512)



Gambar 2.21. Arsitektur Jaringan FMN III

Pada lapisan (B), bobot-bobot hasil pembelajaran dijumlahkan. Kemudian dihitung menggunakan fungsi aktivasi pada lapisan (C). Selanjutnya dihasilkan nilai keanggotaan antesenden pada lapisan (E). Pada bagian konsekuen, hasil pembelajaran adalah nilai inferensi yang dihitung menggunakan persamaan linear. Hasil ini diperoleh pada lapisan (J). Pada lapisan (K), *output* setiap *cluster* dihitung menggunakan Persamaan (2.57). Kemudian dihitung *output* sistem pada lapisan (L).

BAB III
PEMBAHASAN

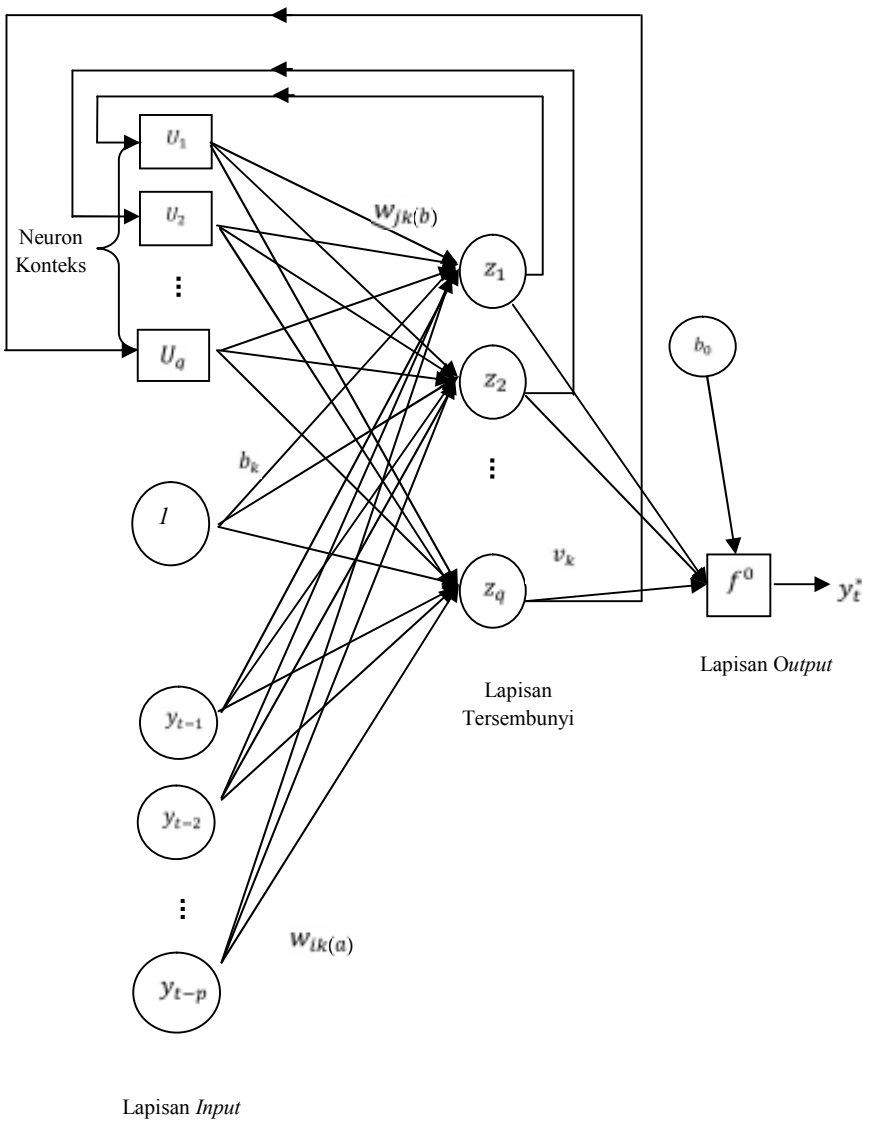
A. Prosedur Pemodelan *Recurrent Neural Network*

Feedforward neural network (FNN) dan *multilayer perceptron* (MLP) adalah jenis jaringan yang umum digunakan dalam pemodelan *neural network*. FNN telah banyak diaplikasikan untuk meramalkan atau klasifikasi data selama beberapa tahun yang lalu. Salah satu tipe dari *neural network* yang dikembangkan dari dasar pemikiran pada FNN. Jaringan tersebut adalah *recurren neural network* atau yang disingkat RNN.

1. Konsep Dasar *Recurrent Neural Network*

Recurrent neural network (RNN) yang juga disebut jaringan umpan balik adalah jenis jaringan pada *neural network* dimana terdapat *loop* sebagai koneksi umpan balik dalam jaringan (Lin & Lee, 1996: 340). Jaringan RNN adalah jaringan yang mengakomodasi *output* jaringan untuk menjadi *input* pada jaringan tersebut yang kemudian digunakan untuk menghasilkan *output* yang baru. Jaringan RNN lebih kompleks jika dibandingkan dengan jaringan FNN. RNN merupakan jaringan dengan kemampuan dinamis karena perilaku jaringan tidak hanya bergantung pada *input* saat ini saja melainkan pada operasi sebelum jaringan.

Terdapat 2 macam jaringan RNN yaitu Jaringan Elman dan Jaringan Hopfield. Pada tugas akhir ini, digunakan jaringan RNN Elman untuk peramalan, sehingga pembahasan akan dikhususkan pada RNN jaringan Elman. Jaringan Elman termasuk pada jaringan *recurrent* sederhana karena mempunyai koneksi umpan balik yang hanya terdapat pada lapisan tersembunyi sedangkan jaringan Hopfield mempunyai koneksi umpan balik pada seluruh neuron yang terbangun. Pada jaringan Elman, hanya terdapat 1 lapisan tersembunyi. Untuk data *time series, input* jaringan Elman adalah *lag* yang signifikan dari data. Gambar 3.1 adalah arsitektur dari jaringan Elman untuk data *time series*



Gambar 3.1. Arsitektur Jaringan Elman Data *Time Series*

Model matematis untuk jaringan Elman adalah sebagai berikut

$$y_t^* = \sum_{k=1}^{q} v_k \frac{1-\exp\left(-\left(\sum_{i=1}^{p} y_{t-i} w_{ik(a)} + \sum_{j=1}^{q} U_j w_{jk(b)} + b_k\right)\right)}{1+\exp\left(-\left(\sum_{i=1}^{p} y_{t-i} w_{ik(a)} + \sum_{j=1}^{q} U_j w_{jk(b)} + b_k\right)\right)} + b_0 + \varepsilon \qquad (3.1)$$

dengan

y^* : variabel *output*

y_{t-i} : variabel *input, i* = adalah *lag* yang signifikan dari plot ACF

z_k : neuron lapisan tersembunyi, *k*=1,2,3…*q*

U_j : variabel *input* yang berada pada neuron tambahan (*neuron konteks*) dengan *j* =1,2,..,*q*

b_k : bobot bias pada neuron ke-*k* pada lapisan tersembunyi dengan *k*=1,2,.., *q*

b_0 : bobot bias pada neuron lapisan *output*

$w_{ik(a)}$: bobot dari lapisan *input* ke-*i* menuju neuron ke-*k* pada lapisan tersembunyi

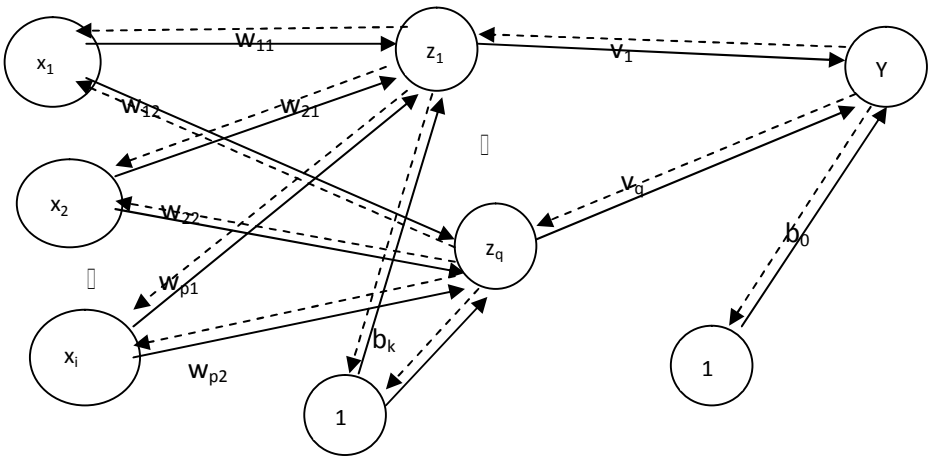
$w_{jk(b)}$: bobot dari lapisan *input* (neuron tambahan) ke-*j* menuju neuron ke-*k* pada lapisan tersembunyi

v_k : bobot dari neuron ke-*k* pada lapisan tersembunyi yang menuju lapisan *output* dengan *k* =1,2,.., *q*.

ε : *error*

2. Algoritma Pembelajaran *Backpropagation*

Tugas akhir ini menggunakan algoritma *backpropagation* sebagai algoritma pembelajaran. Algoritma *backpropagation* termasuk dalam algoritma terawasi. Algoritma ini terdiri dari 2 tahap yaitu perambatan maju (*forward propagation*) dan perambatan mundur (*backward*). Perambatan maju dikerjakan untuk mendapatkan *error*. Kemudian *error* yang didapatkan digunakan untuk mengubah nilai-nilai bobotnya dalam arah mundur. Pada saat perambatan maju, *neuron-neuron* diaktifkan menggunakan fungsi aktivasi yang dapat didiferensialkan. Arsitektur jaringan *backpropagation* terlihat pada Gambar 3.2.



Gambar 3.2. Arsitektur Jaringan *Backpropagation*

pada Gambar 3.2

x_i	: *input, i*=1,2,…*p*
z_j	: *neuron* pada lapisan tersembunyi, *j*=1,2,….*q*
Y	: *output*
w_{ij}	: bobot-bobot yang menghubungkan antara *neuron*-*neuron* pada lapisan *input* dengan lapisan tersembunyi
v_k	: bobot-bobot yang menghubungkan antara *neuron*-*neuron* pada lapisan tersembunyi dengan lapisan *output.*
b_k	: bobot bias yang menuju ke lapisan tersembunyi
b_0	: bobot bias yang menuju lapisan *output*.

Algoritma *backpropagation* secara umum adalah sebagai berikut:

a. Inisialisasi bobot (ambil bobot awal dengan nilai random yang cukup kecil)

b. Tetapkan parameter, parameter yang ditetapkan adalah sebagai berikut:

1) Maksimum epoch

Maksimum epoch adalah jumlah epoch maksimum yang boleh dilakukan selama proses pembelajaran. Iterasi akan dihentikan apabila nilai epoch melebihi maksimum epoch.

Perintah: net.trainParam.epochs = MaxEpoch

Nilai default untuk maksimum epoch adalah 10.

2) Kinerja tujuan

Kinerja tujuan adalah target nilai fungsi kinerja. Iterasi akan dihentikan apabila nilai fungsi kinerja kurang dan atau sama dengan kinerja tujuan.

Perintah: net.train.Param.goal = TargetError

Nilai default untuk kinerja tujuan adalah 0.

3) *Learning rate*

Learning rate adalah laju pembelajaran. Semakin besar nilai *learning rate*, semakin besar pula langkah pembelajaran. Semakin kecil *learning rate*, maka proses pembelajaran akan sangat lama. Sehingga perlu pemilihan nilai yang tepat untuk *learning rate*.

Perintah: net.trainParam.lr = LearningRate

Nilai default untuk *learning rate* adalah 0.01.

4) Rasio untuk menaikkan *learning rate*

Rasio ini berguna sebagai faktor pengali untuk menaikkan *learning rate* apabila *learning rate* yang ada terlalu rendah untuk mencapai kekonvergenan.

Perintah : net.trainParam.lr.inc=IncLearningRate

Nilai default untuk rasio kenaikan *learning rate* adalah 1.05.

5) Rasio untuk menurunkan *learning rate*

Rasio ini berguna sebagai faktor pengali untuk menurunkan *learning rate* apabila learning rate yang ada terlalu tinggi untuk menuju ketidakstabilan.

Perintah : net.trainParam.lr.dec=DecLearningRate

Nilai default untuk rasio penurunan *learning rate* adalah 0.7.

6) Jumlah epoch yang akan ditunjukkan kemajuannya

Menunjukkan jumlah epoch akan ditampilkan.

Perintah: net.trainParam.show = EpochShow

Nilai default untuk jumlah epoh yang akan ditunjukkan adalah 25.

7) Maksimum kenaikan kinerja

Maksimum kenaikan kinerja adalah nilai maksimum kenaikan *error* yang diijinkan, antara *error* saat ini dan *error* sebelumnya.

Perintah: net.trainParam.max_perf_inc=MaxPerfInc

Nilai default untuk maksimum kenaikan kinerja adalah 1,04.

8) Momentum

Momentum adalah perubahan bobot yang didasarkan atas arah gradient pola terakhir dan pola sebelumnya. Besarya momentum antara 0 sampai 1. Apabila nilai momentum = 0, maka perubahan bobot hanya akan dipengaruhi oleh gradiennya. Tetapi, apabila nilai momentum = 1, maka perubahan bobot akan sama dengan perubahan bobot sebelumnya.

Perintah: net.trainParam.mc = Momentum

c. Kerjakan langkah-langkah berikut selama (Epoch < Maksimum Epoch) dan (MSE < Target *Error*):

Epoch = Epoch + 1

Untuk tiap-tiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan:

Feedforward:

1) Tiap-tiap neuron *input* (x_i, $i=1,2,3...p$) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua neuron pada lapisan setelahnya, yaitu lapisan tersembunyi (*hidden layer*).
2) Tiap-tiap neuron pada lapisan tersembunyi (z_j, $j=1,2,...q$) menjumlahkan sinyal-sinyal *input* terbobot sebagai berikut:

$$z\_in_j = b_k + \sum_1^p x_i w_{ik(\alpha)}$$

b_k= bias pada neuron tersembunyi

untuk menghitung sinyal *output*nya digunakan fungsi aktivasi sebagai berikut:

$$z_j = f(z\_in_j)$$

kemudian sinyal tersebut dikirim ke semua neuron di lapisan *output* dan neuron tambahan di lapisan *input*. Langkah ini dilakukan sebanyak jumlah lapisan tersembunyi.

3) Tiap-tiap neuron tersembunyi meneruskan sinyal pada neuron tambahan pada lapisan *input* U_k, $k=1, 2, 3,\ldots\ldots q$ dan jumlahkan bobot sinyal *inputnya*. Terdapat dua sinyal *input* dari neuron tambahan yaitu:

   a) Sinyal *output* neuron tambahan ke lapisan tersembunyi:

      $$U\_net_q = b_k + \sum_{i=1}^{p} x_i w_{jk(b)}$$

      Digunakan fungsi aktivasi untuk menghitung sinyal *outputnya*:

      $$U_k = f(U\_net_q)$$

      Kemudian kirimkan sinyal tersebut ke lapisan tersembunyi.

   b) Sinyal *output* neuron tambahan ke lapisan *output*:

      $$T\_in_k = b_k + \sum_{i=1}^{p} x_i w_{ik(a)} + \sum_{i=1}^{q} u_j w_{jk(b)}$$

      Digunakan fungsi aktivasi untuk menghitung sinyal *outputnya*:

      $$T_k = f(T\_net_q)$$

      kemudian kirimkan sinyal tersebut ke semua neuron pada lapisan *output*.

4) Tiap-tiap neuron *output* (*Y*) menjumlahkan sinyal-sinyal *input* terbobot menggunakan rumus.

   $$y\_in = b_0 + \sum_{i=1}^{p} v_{k1} T_k$$

   dengan, b_0= bias pada neuron *output l.*

   untuk menghitung sinyal *output*nya digunakan fungsi aktivasi sebagai berikut:

   $$y = f(y\_in)$$

   kemudian sinyal tersebut dikirim ke semua neuron di lapisan atasnya yaitu neuron-neuron *output*.

Backpropagation

1) Tiap-tiap neuron *output* (y) menerima target pola yang berhubungan dengan pola *input* pembelajaran, kemudian dihitung informasi *error*nya:

   $$\boldsymbol{\delta_k = (t - y)f'(y\_in)}$$

   kemudian dihitung koreksi bobot yang nantinya akan digunakan untuk memperbaiki nilai v_k dengan rumus sebagai berikut:

   $$\Delta v_k = \alpha \delta_k z_j$$

selain itu, menghitung koreksi bias yang nantinya akan digunakan untuk memperbaiki nilai b_0, dan kirimkan δ_k ke neuron-neuron pada lapisan sebelumnya dengan rumus

$$\Delta b_0 = \alpha \delta_k$$

2) Tiap-tiap neuron tersembunyi ($z_j, j = 1,2, \dots q$) menjumlahkan delta *input*nya dari neuron-neuron yang berada pada lapisan di atasnya sebagai berikut:

$$\delta\_in_j = \sum_{k=1}^{q} \delta_k v_k$$

untuk menghitung informasi *error,* kalikan nilai δ_in_j dengan turunan dari fungsi aktivasinya:

$$\delta_j = \delta\_in_j f'(z\_in_j)$$

kemudian dihitung koreksi bobot yang nantinya akan digunakan untuk memperbaiki nilai $w_{ik(a)}$ dan $w_{jk(b)}$ dengan rumus sebagai berikut:

$$\Delta w_{ik(a)} = \alpha \delta_j x_i$$

$$\Delta w_{jk(b)} = \alpha \delta_j U_q$$

selain itu, menghitung koreksi bias yang nantinya akan digunakan untuk memperbaiki nilai b_k:

$$\Delta b_k = \alpha \delta_j$$

3) Tiap-tiap neuron *output*($\bar{y}_k^*$) memperbaiki bias dan bobotnya sebagai berikut:

$$v_k(baru) = v_k(lama) + \Delta v_k$$

$$\boldsymbol{b_0(baru) = b_0(lama) + \Delta b_0}$$

Tiap-tiap neuron lapisan tersembunyi ($z_j, j = 1,2, \dots q$) memperbaiki bias dan bobotnya sebagai berikut:

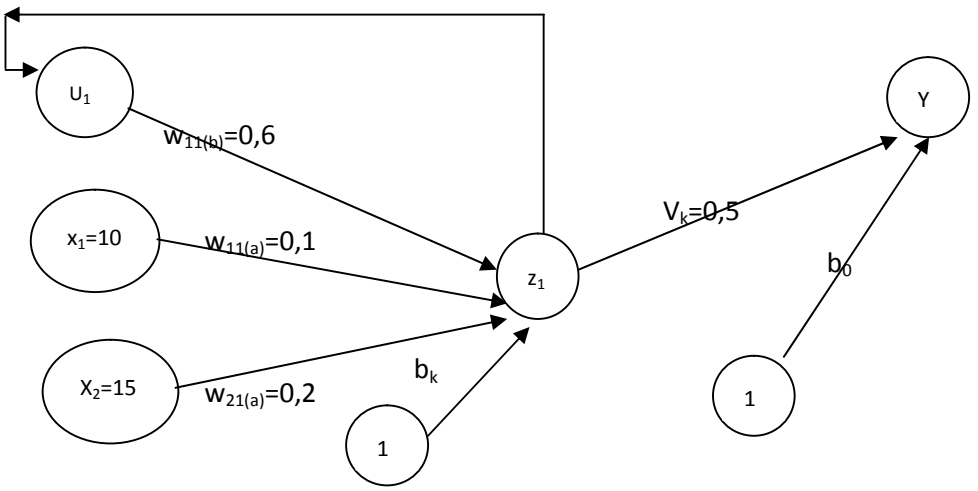
$$\boldsymbol{w_{ik(a)}(baru) = w_{ik(a)}(lama) + \Delta w_{ik(a)}}$$

$$\boldsymbol{w_{jk(b)}(baru) = w_{jk(b)}(lama) + \Delta w_{jk(b)}}$$

$$\boldsymbol{b_k(baru) = b_k(lama) + \Delta b_k}$$

Contoh 3.1. Perbaikan bobot menggunakan algoritma *backpropagatioan*.

Misalkan jaringan Elman dengan 2 *input* dan 1 neuron tersembunyi seperti pada Gambar 3.3. Akan dihitung bobot-bobot jaringan menggunakan fungsi aktivasi linear dengan α= 0,01 dan target *output*=6.



Gambar 3.3. Arsitektur Jaringan

1) Tiap-tiap neuron *input (xi, i=1,2)* menerima sinyal x_i dan meneruskan sinyal tersebut ke semua neuron pada lapisan setelahnya, yaitu lapisan tersembunyi (*hidden layer*).
2) Neuron pada lapisan tersembunyi (z_1) menjumlahkan sinyal-sinyal *input* terbobot sebagai berikut:

$z_{in_j} = b_k + \sum_1^p x_i w_{ik(a)} = b_k + x_1 w_{11(a)} + x_2 w_{21(a)} = 1 + (10 \times 0{,}1) + (15 \times 0{,}2) = 5$

untuk menghitung sinyal *output*nya digunakan fungsi aktivasi linear ($y=x$) sebagai berikut:

$z_j = f(z\_in_j)$

$\mathbf{z_j = z_{in_j} = 5}$

kemudian sinyal tersebut dikirim ke semua neuron di lapisan *output* dan neuron tambahan di lapisan *input*.

3) Tiap-tiap neuron tersembunyi meneruskan sinyal pada neuron tambahan pada lapisan *input* U_k, $k=1, 2, 3,\ldots\ldots q$ dan jumlahkan bobot sinyal *inputnya*. Terdapat dua sinyal *input* dari neuron tambahan yaitu:

   a) Sinyal *output* neuron tambahan ke lapisan tersembunyi:

      $U_{net_q} = b_k + \sum_{i=1}^{p} x_i w_{jk(b)} = 1 + (5 \times 0{,}6) = 1 + 3 = 4$

      Digunakan fungsi aktivasi untuk menghitung sinyal *outputnya*:

      $U_k = f(U\_net_q)$

      $U_k = U_{net_q} = 4$

      Kemudian kirimkan sinyal tersebut ke lapisan tersembunyi.

   b) Sinyal *output* neuron tambahan ke lapisan *output*:

      $T\_in_k = b_k + \sum_{i=1}^{p} x_i w_{ik(a)} + \sum_{i=1}^{q} u_j w_{jk(b)}$

      $T\_in_k = 1 + (10 \times 0{,}1) + (15 \times 0{,}2) + (5 \times 0{,}6) = 8$

      Digunakan fungsi aktivasi untuk menghitung sinyal *outputnya*:

      $T_k = f(T\_net_q)$

      $T_k = T\_in_k = 8$

      kemudian kirimkan sinyal tersebut ke semua neuron pada lapisan *output*.

4) Tiap-tiap neuron *output* (*Y*) menjumlahkan sinyal-sinyal *input* terbobot menggunakan rumus.

   $y\_in = b_0 + \sum_{i=1}^{p} v_{k1} T_k = 1 + (0{,}5 \times 8) = 5$

   untuk menghitung sinyal *output*nya digunakan fungsi aktivasi sebagai berikut:

   $y = f(y_{in})$

   $y = y\_in = 5$

   kemudian sinyal tersebut dikirim ke semua neuron di lapisan atasnya yaitu neuron-neuron *output*.

Backpropagation

1) Tiap-tiap neuron *output* (y) menerima target pola yang berhubungan dengan pola *input* pembelajaran, kemudian dihitung informasi *error*nya:

$$\boldsymbol{\delta_k = (t - y)f'(y_{in}) = (6 - 5) \times 1 = 1}$$

kemudian dihitung koreksi bobot yang nantinya akan digunakan untuk memperbaiki nilai v_k dengan rumus sebagai berikut:

$\Delta v_k = \alpha \delta_k z_j = 0{,}01 \times 1 \times 5 = 0{,}05$

selain itu, menghitung koreksi bias yang nantinya akan digunakan untuk memperbaiki nilai b_0, dan kirimkan δ_k ke neuron-neuron pada lapisan sebelumnya dengan rumus

$\Delta b_0 = \alpha \delta_k = 0{,}01 \times 1 = 0{,}01$

2) Neuron tersembunyi menjumlahkan delta *input*nya dari neuron-neuron yang berada pada lapisan di atasnya sebagai berikut:

$\delta\_in_j = \sum_{k=1}^{q} \delta_k v_k = 1 \times 0{,}5 = 0{,}5$

untuk menghitung informasi *error,* kalikan nilai δ_in_j dengan turunan dari fungsi aktivasinya:

$$\delta_j = \delta\_in_j f'(z\_in_j)$$

$\delta_j = 0{,}5 \times 1 = 0{,}5$

kemudian dihitung koreksi bobot yang nantinya akan digunakan untuk memperbaiki nilai $w_{ik(a)}$ dan $w_{jk(b)}$ dengan rumus sebagai berikut:

$\Delta w_{ik(a)} = \alpha \delta_j x_i$

$\Delta w_{1k(a)} = \alpha \delta_j x_1 = 0{,}01 \times 0{,}5 \times 10 = 0{,}05$

$\Delta w_{2k(a)} = \alpha \delta_j x_2 = 0{,}01 \times 0{,}5 \times 15 = 0{,}075$

$\Delta w_{jk(b)} = \alpha \delta_j U_q = 0{,}01 \times 0{,}5 \times 4 = 0{,}02$

selain itu, menghitung koreksi bias yang nantinya akan digunakan untuk memperbaiki nilai b_k:

$\Delta b_k = \alpha \delta_j = 0{,}01 \times 0{,}5 = 0{,}005$

3) Tiap-tiap neuron *output*(y_k^*) memperbaiki bias dan bobotnya sebagai berikut:

$v_k(baru) = v_k(lama) + \Delta v_k = 0{,}5 + 0{,}05 = 0{,}55$

$\boldsymbol{b_0(baru) = b_0(lama) + \Delta b_0 = 1 + 0{,}01 = 1{,}01}$

Neuron lapisan tersembunyi memperbaiki bias dan bobotnya sebagai berikut:

$\boldsymbol{w_{ik(a)}(baru) = w_{ik(a)}(lama) + \Delta w_{ik(a)}}$

$\boldsymbol{w_{1k(a)}(baru) = w_{1k(a)}(lama) + \Delta w_{1k(a)} = 0{,}1 + 0{,}05 = 0{,}15}$

$\boldsymbol{w_{2k(a)}(baru) = w_{2k(a)}(lama) + \Delta w_{2k(a)} = 0{,}2 + 0{,}075 = 0{,}275}$

$\boldsymbol{w_{jk(b)}(baru) = w_{jk(b)}(lama) + \Delta w_{jk(b)} = 0{,}6 + 0{,}02 = 0{,}62}$

$\boldsymbol{b_k(baru) = b_k(lama) + \Delta b_k = 1 + 0{,}005 = 1{,}005}$

Demikian seterusnya hingga mencapai *error* yang diinginkan.

3. Membangun Jaringan Elman

Seperti yang telah dijelaskan bahwa jaringan Elman memiliki 1 lapisan tersembunyi. Setiap lapisan akan menerima bobot dari lapisan sebelumnya. Semua lapisan kecuali lapisan terakhir memiliki satu bobot yang disebut bobot *recurrent*. Setiap lapisan memiliki bias. Peramalan akan menggunakan bantuan *Matlab R2009a*. Pada toolbox *matlab*, perintah untuk membentuk jaringan adalah sebagai berikut:

net = newelm (PR, [S1 S2 ...SN1], {TF1 TF2 ...TFN1}, BTF, BLF, PF)

dengan

PR : matriks berukuran Rx2 yang berisi nilai minimum dan maksimum, dengan R adalah jumlah variabel *input*.

Si : jumlah neuron pada lapisan ke-*l*

TFi : fungsi aktivasi pada lapisan ke-*l*, dengan *l*= 1,2,..,*l* (default: *tansig*)

BTFi : fungsi pembelajaran untuk jaringan *backpropagation* (default: *traingdx*). Dapat berupa fungsi yang lain seperti: *traingd, traingdm, traingda, traingdx,* dll.

BLFi : fungsi pembelajaran untuk perbaikan bobot-bobot pada *backpropagation* (default: *learngdm).*

PF : fungsi kinerja (default: mse).

4. Prosedur Pemodelan Jaringan Elman

Pada tugas akhir ini, jaringan yang akan digunakan untuk peramalan adalah jaringan Elman. Jaringan Elman adalah jaringan dengan 3 lapisan yaitu lapisan *input*, 1 lapisan tersembunyi, dan lapisan *output* dengan tambahan neuron pada lapisan *input* yang berasal dari lapisan tersembunyi. Jaringan Elman digunakan untuk pembelajaran data dengan menggunakan algorima *backpropagation*. Berikut prosedur pemodelan jaringan Elman.

a. Penentuan *input* jaringan

Langkah awal pemodelan menggunakan jaringan Elman adalah penentuan *input*. *Input* jaringan diperoleh dengan melihat plot autokorelasi data. Variabel-variabel yang layak dijadikan *input* jaringan adalah *lag-lag* signifikan pada fungsi autokorelasi data. Misalkan *lag-lag* signifikan adalah *lag* 1 dan *lag* 13, maka *input* yang digunakan adalah data y_{t-1} dan data y_{t-13}. Sedangkan *output*nya adalah data saat *t*.

b. Pembagian data

Setelah penentuan *input*, langkah selanjutnya adalah pembagian data. Dalam pembagian data, data dibagi menjadi data pembelajaran (*training)*, dan data pengujian (*testing)*. Pembagian data yang dapat digunakan adalah 75% untuk data *training* dan 25% untuk data *testing* (Lin et al, 2008:1). Aspek pembagian data harus ditekankan supaya memperoleh data *training* yang secukupnya yang selanjutnya digunakan untuk proses pembelajaran dan data *testing* digunakan untuk pengujian. Proses pembelajaran yang dilakukan data *training* berdasarkan nilai *MSE* dan *MAPE* data *training* dan *testing*.

c. Normalisasi Data

Proses normalisasi dapat dilakukan dengan beberapa cara. Cara tersebut antara lain dengan meletakkan data-data *input* dan target pada *range* tertentu. Selain itu, proses normalisasi juga dapat dilakukan dengan bantuan *mean* dan standar deviasi (Kusumadewi, 2004: 191). Dalam tugas akhir ini menggunakan bantuan *mean* dan standar deviasi untuk normalisasi data. Perhitungan normalisasi dilakukan dengan rumus:

$$n_i = \frac{y_i - \bar{y}}{s} \tag{3.2}$$

dengan

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

adalah rata-rata data dan

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

adalah nilai variansi data.

Fungsi *prestd* dalam *matlab* akan membawa data ke dalam bentuk normal yaitu data dengan *mean*=0 dan standart deviasi= 1. Perintah matlab untuk fungsi *prestd* adalah

[pn, meanp, stdp, tn, meant, stdt] = prestd (P,T)

dengan

P : matriks *input*

T : matriks target

pn : matriks *input* yang ternormalisasi

tn : matriks target yang ternormalisasi

maenp : mean pada matriks *input* asli

stdp : standart deviasi pada matriks *input* asli

meant : mean pada matriks target asli

stdt : standart deviasi pada matriks target asli

d. Pembentukan Model Terbaik

Pembentukan model terbaik meliputi dua langkah penting yaitu penentuan jumlah neuron lapisan tersembunyi dan melakukan eliminasi variabel *input*. Penentuan jumlah neuron lapisan tersembunyi dilakukan dengan cara *trial* dan *error* beberapa model yang mungkin, kemudian membandingkan nilai MSE dan MAPE model-model yang dibangun. Menggunakan prinsip parsimoni, model terbaik adalah model dengan MSE dan MAPE terkecil dengan jaringan paling sederhana. Setelah mendapatkan jumlah neuron tersembunyi, langkah selanjutnya adalah eliminasi *input*. Masing-masing *input* dikeluarkan dari dalam model kemudian memilih model terbaik. Pemilihan model terbaik dan eliminasi *input* dilakukan dengan melakukan pembelajaran terhadap model-model yang mungkin menggunakan algoritma pembelajaran *backpropagation*.

Seperti halnya pada jaringan *feedfoorward*, pembelajaran dilakukan dalam rangka melakukan pengaturan bobot. Pengaturan ini dimaksudkan untuk memperoleh bobot-bobot, MAPE, dan MSE yang baik pada akhir pembelajaran. Selama proses pembelajaran, bobot-bobot diatur secara iteratif untuk meminimumkan fungsi kinerja jaringan. Fungsi kinerja yang sering digunakan untuk *backpropagation* adalah MSE, nilai MSE dihitung dari rata-rata kuadrat *error* yang terjadi antara *output* jaringan dan target. Pada matlab, terdapat 2 cara untuk pengimplementasian algoritma *gradient descent*, yaitu dengan *incremental mode* dan *batch mod*e. Pada tugas akhir ini, akan digunakan *batch mode* untuk perhitungan gradien. Pada *batch mode*, perhitungan gradien dan perbaikan nilai bobot-bobot dilakukan setelah pengoperasian semua *input* data. Dalam matlab, untuk pembelajaran *backpropagation* dengan *batch mode* menggunakan fungsi train dengan perintah

[net,tr] = train(net,P,T,Pi,Ai)

dengan

net : jaringan syaraf

tr : informasi pembelajaran (epoch dan fungsi kinerja)

P : matriks data *input*

T : matriks data target (default: 0)

Pi : kondisi awal *delay input* (default: 0)

Ai : kondisi awal *delay* lapisan (default: 0).

Fungsi train akan menggunakan objek jaringan, kumpulan data *input*, dan target sebagai *input* pembelajaran. Fungsi ini akan menghasilkan objek jaringan terlatih, bobot-bobot akhir, dan informasi selama pembelajaran yaitu epoh dan fungsi kinerja sebagai nilai *output*. Fungsi pembelajaran untuk bobot-bobot menggunakan *gradient descent* antara lain *gradient descent* (traingd) dan *gradient descent* dengan momentum (traingdm). Fungsi *gradient descent* (traingd) diperbaiki menjadi fungsi *gradient descent* dengan momentum dan *adaptive learning rate* (traingdx). Pada tugas akhir ini akan digunakan fungsi traingdx.

e. Denormalisasi

Denormalisasi adalah pengembalian ke dalam bentuk semula terhadap data yang telah dinormalisasi. Karena pada awal pembelajaran telah dilakukan normalisasi pada data asli, maka *output* jaringan memiliki *mean*=0 dan standar deviasi=1. Sehingga perlu membawa *output* jaringan tersebut sesuai dengan kondisi data asli. Fungsi pada *matlab* untuk proses denormalisasi adalah fungsi poststd dengan perintah:

[P,T]= poststd(pn, meanp, stdp, tn, meant, stdt)

dengan

P : matriks *input*

T : matriks target

Pn : matriks *input* yang ternormalisasi

tn : matriks target yang ternormalisasi

maenp : *mean* pada matriks *input* asli

stdp : standar deviasi pada matriks *input* asli

meant : *mean* pada matriks target asli

stdt : standart deviasi pada matriks target asli

Sebelum melakukan proses denormalisasi, perlu dilakukan simulasi pada jaringan syaraf dengan perintah

an = sim(net,pn);

a = poststd(an,meant,stdt);

f. Uji Kesesuain Model

Uji kesesuain model dilakukan setelah mendapatkan hasil peramalan menggunakan model terbaik. Uji ini untuk melihat apakah model layak atau tidak untuk digunakan sebagai model peramalan. Uji ini dilakukan dengan melihat kriteria *white noise* dari *error* peramalan pada plot ACF dan plot PACF *error* model.

B. Prosedur Pemodelan *Recurrent Neuro Fuzzy*

Pada bagian ini, akan dibahas tentang realisasi *neural network* dari sistem inferensi *fuzzy* Sugeno. Ide dasar menggunakan *neural network* untuk menggeneralisasi model Soegeno adalah untuk membangkitkan fungsi keanggotaan pada bagian antesenden. Disamping digunakan untuk membangkitkan fungsi keanggotaan, *neural network* juga digunakan sebagai fungsi pembelajaran pada inferensi *fuzzy*. Takagi dan Hayasi akan menggunakan *neural network* dengan algoritma pembelajaran *backpropagation* untuk membangun himpunan-himpunan *fuzzy* pada bagian antesenden serta fungsi inferensi pada bagian konsekuen. Aturan inferensi yang digunakan adalah sebagai berikut (Lin & Lee, 1996:507)

$$R^s: IF\ x = (x_1, x_2, .. x_p) is\ A_s\ THEN\ y_s = NN_s(x_1, x_2, .. x_p); s = 1,2 \dots, r \quad (3.3)$$

dengan *r* adalah jumlah aturan inferensi, A_s himpunan *fuzzy* bagian antesenden pada setiap aturan, dan NN_s adalah jaringan *backpropagation* dengan *input* $x_1, x_2, .. x_p$ dan y_s adalah *output* jaringan. Pada tugas akhir ini, pembelajaran menggunakan *recurrent neural network* jaringan Elman. Sehingga untuk selanjutnya, penyebutan model *neuro fuzzy* diganti dengan model *recurrent neuro fuzzy*. Arsitektur jaringan *recurrent neuro fuzzy* terlihat pada Gambar 2.21. Berikut adalah langkah-langkah pembentukan sistem inferensi *fuzzy*

model Sugeno melalui pengendali *recurrent neural network* (Lin & Lee. 1996:509-510):

1. Penentuan *input* jaringan

   Langkah awal pemodelan menggunakan *recurrent neuro fuzzy* adalah penentuan *input* jaringan. Karena pada tugas akhir ini variabel *input* yang digunakan adalah variabel data *time series,* maka *input* jaringan adalah *lag-lag* yang signifikan pada plot ACF data dengan *output* data saat *t*. Misalkan didefinisiskan variabel *output* adalah *y* dan kandidat untuk variabel *input* adalah x_i dengan *i=1, 2, ...p*. Setelah *input* ditentukan, variabel x_i yang berhubungan dengan nilai *output* y_i diseleksi menggunakan algoritma *backpropagation* pada *recurrent neural network*. Hal ini dilakukan dengan menggunakan metode eliminasi *backward*. Pemilihan didasarkan oleh nilai MAPE dan MSE data. Jika eliminasi *input* pada variabel *input* tertentu memberikan penurunan yang besar terhadap nilai MAPE dan MSE model, maka *input* yang bersangkutan akan dieliminasi.
2. Pembagian data

   Setelah penentuan *input*, data dibagi menjadi data pembelajaran (*training)*, dan data pengujian (*testing)*. Pembagian data yang dapat digunakan adalah 75% untuk data *training* dan 25% untuk data *testing* (Lin et al, 2008:1). Aspek pembagian data harus ditekankan supaya memperoleh data *training* yang secukupnya yang selanjutnya digunakan untuk proses pembelajaran dan data *testing* digunakan untuk pengujian.
3. Pengelompokan (*clustering*) data pembelajaran

   Pada langkah ini, data *training* akan dikelompokkan menjadi *r cluster*. Pengelompokan menggunakan metode yang dipilih. Dalam tugas akhir ini, proses *clustering* menggunakan *Fuzzy C-Means,* sehingga jaringan akan memiliki *r* buah aturan R^s dengan *s=1,2,3,..r.* Pasangan *input-output* direpresentasikan sebagai (x_i^s, y_i^s), dengan *i=1,2,3…….*N_s, N_s adalah banyak data yang masuk dalam *cluster* ke-*s*.

4. Pembelajaran *recurrent neural network* yang berhubungan dengan bagian antesenden (bagian IF) pada aturan-aturan inferensi *fuzzy*

   Untuk masing-masing *input* pada data pembelajaran, didefinisikan $m_i = \left(m_i^1, m_i^2, ..., m_i^r\right), i = 1,2,3,..., N_t$, dengan ketentuan

$$m_i^k = \begin{cases} 1; & k = s \\ 0; & k \neq s \end{cases} \tag{3.4}$$

   dengan

   N_t : banyak pasangan data *training*

   Nilai keanggotaan setiap data bagian IF didefinisikan dengan

$$\mu_{A_S}(x_i) = \hat{m}_i^s; \; i = 1,2,...,N; s = 1,2,...,r \tag{3. 5}$$

   dengan A_s adalah himpunan *fuzzy* hasil prekondisi s aturan yang telah didefinisikan.

5. Pembelajaran *recurrent neural network* yang berhubungan dengan bagian konsekuen (bagian THEN) pada aturan-aturan inferensi *fuzzy*

   Input data *training* $x_{i1}^s, x_{i2}^s, \dots x_{im}^s$ dan nilai *output* $y_i^s, i = 1,2, \dots, N_s$ dijadikan *input-output* dari NN_s. Dalam hal ini NN_s adalah *neural network* bagian THEN. Hasil pembelajaran akan diujikan pada data *testing*. *Input* yang digunakan adalah $x = (x_{i1}^s, x_{i2}^s, \dots, x_{im}^s)$, $i = 1,2,\dots,N_c$. Pengujian ini dilakukan untuk mendapatkan SSE yang didefinisikan sebagai berikut (Lin & Lee, 1996:510)

$$\boldsymbol{E_m^s = \sum_{i=1}^{N_c} [y_i - \mu_s(x_i)\mu_{As}(x_i)]^2} \tag{3.6}$$

   untuk $s = 1,2, \dots, r$ dengan estimasi $\mu_s(x_i)$ diperoleh dari *output* jaringan. *Error* dengan pembobotan dihitung dengan persamaan sebagai berikut (Lin & Lee, 1996:509)

$$\boldsymbol{E_m^s = \sum_{i=1}^{N_c} \mu_{As}(x_i)[y_i - \mu_s(x_i)\mu_{As}(x_i)]^2} \tag{3.7}$$

   dengan:

   s : banyak aturan *inferensi fuzzy*

   y_i : target *output* ke-*i*

N_c : banyak pasangan data *testing*

$\mu_{As}(x_i)$: nilai keanggotaan tiap x_i dalam himpunan *fuzzy A* pada aturan *fuzzy* ke-*s* bagian anteseden

$f_s(x_i)$: *output* jaringan hasil pembelajaran tiap x_i pada setiap aturan inferensi *fuzzy* R^s pada bagian konsekuen

Karena pada tugas akhir ini menggunakan MSE sebagai kriteria kebaikan model, maka hasil pembelajaran pada langkah ini adalah nilai MSE yang didapatkan dengan cara membagi nilai SSE dengan banyak pengamatan. Selain nilai MSE *testing*, penelitian ini juga mempehitungkan nilai MSE *training* hasil pembelajaran.

6. Penyederhanaan bagian konsekuen (bagian THEN) menggunakan metode eliminasi *backward*

   Dari *m input* NN_s tidak semuanya memiliki kontribusi yang cukup baik. Oleh karena itu, salah satu variabel dapat dieliminasi dan kemudian melatih jaringan kembali seperti langkah 3 untuk mendapatkan MSE dan MAPE data. Disamping itu, akan ditentukan parameter konsekuen untuk tiap R^s dengan menggunakan metode *Least Square Estimator* (LSE) untuk mengidentifikasi parameter-parameter linearnya. Setelah langkah ini, diperoleh *output* jaringan.

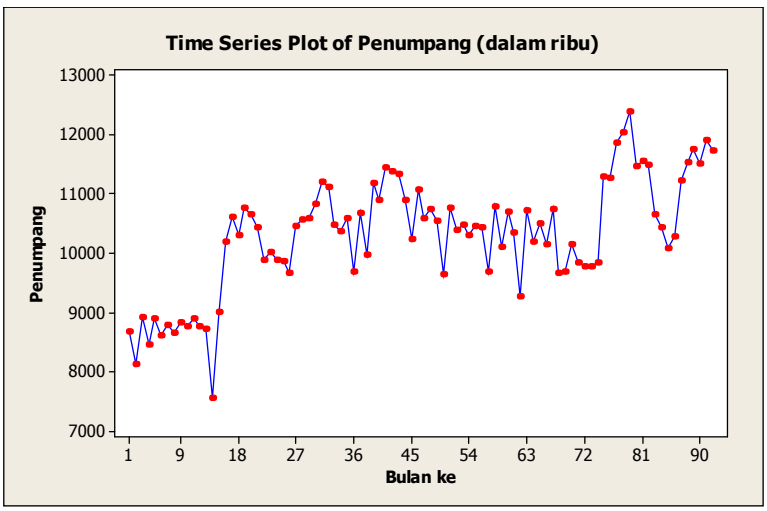
7. Uji Kesesuaian Model

   Uji kesesuain model dilakukan untuk mengetahui apakah model layak atau tidak digunakan sebagai model peramalan. Model yang layak digunakan untuk peramalan adalah model yang memenuhi kriteria *white noise* pada plot ACF dan PACF *error* peramalan.

C. Aplikasi *Recurrent Neural Network* dan *Recurrent Neuro Fuzzy* Jaringan Elman untuk Peramalan Penumpang Kereta Api Jabodetabek

Model *recurrent neural network* dan *recurrent neuro fuzzy* adalah model yang dapat digunakan untuk peramalan data *time series*. Pada tugas akhir ini akan dilakukan pengaplikasian model *recurrent neural network* dan *recurrent neuro fuzzy* untuk permalan jumlah penumpang kereta api Jabodetabek dengan langkah prosedur yang telah diterangkan sebelumnya.

Data penumpang kereta api merupakan data *time series*. Data tersebut merupakan data kuantitatif yang disusun menurut urutan waktu. Data yang akan digunakan untuk peramalan adalah data jumlah penumpang kereta api di daerah Jabodetabek yang diunduh di *www.bps.go.id* pada tanggal 3 Oktober 2013. Data tersebut menunjukkan jumlah penumpang kereta dari periode Januari 2006 sampai Agustus 2013 yang diperinci setiap bulannya. Data lengkap jumlah penumpang kereta api tersebut terlampir pada Lampiran 3.1. Gambar 3.4 menunjukkan pola data jumlah penumpang kereta api Jabodetabek

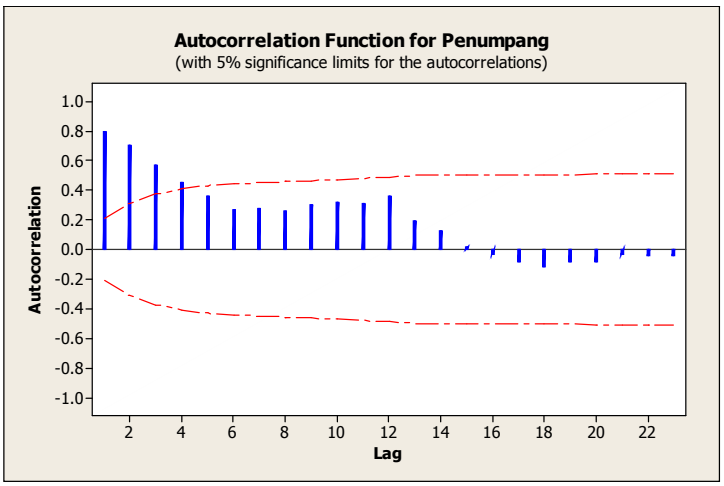


Gambar 3.4. Plot Jumlah Penumpang Kereta

Dari Gambar 3.4 plot data menunjukkan bahwa data cenderung memiliki pola Horisontal. Titik-titik berfluktuasi di sekitar nilai 10000. Terjadi lonjakan yang cukup signifikan dari data ke-16 yaitu penumpang pada bulan Maret 2007. Jumlah penumpang yang semula berfluktuasi di antara nilai 8000 sampai 9000 naik hingga mencapai angka 10000. Lonjakan tersebut dilatar belakangi oleh perbaikan fasilitas kereta api yang terus dilakukan sehingga kereta api menjadi salah satu alternatif alat transportasi pilihan. Selain itu, harga tiket yang terjangkau, terhindarnya dari kemacetan, dan waktu tempuh yang singkat menjadikan kereta api menjadi alat transportasi yang sering digunakan.

Fluktuasi jumlah penumpang mengindikasi bahwa jumlah penumpang menunjukkan jumlah yang beragam setiap bulannya. Kadang kala terjadi

lonjakan yang sangat signifikan yang tidak terprediksi sebelumnya, sehingga menyebabkan banyaknya penumpang kehabisan tiket. Kesulitan prediksi jumlah penumpang kereta api inilah yang menjadikan peramalan adalah hal yang perlu dilakukan.



Gambar 3.5. Plot ACF

Gambar 3.5 Menunjukkan plot ACF penumpang kereta api. Dari plot ACF, *lag* yang signifikan adalah *lag* 1, *lag* 2, *lag* 3, dan *lag* 4. Sehingga dari plot ACF dapat disimpulkan bahwa data tersebut tidak mengalami pola musiman tahunan, 6 bulanan, ataupun kuartalan.

1. Aplikasi *Recurrent Neural Network* untuk Peramalan Jumlah Penumpang Kereta Api Jabodetabek

   Prosedur awal peramalan menggunakan model *recurren neural network* adalah menentukan *input* jaringan. Penentuan *input* dapat dilakukan dengan cara melihat *lag-lag* yang signifikan pada gambar plot fungsi *autokorelasi*. Plot fungsi *autokorelasi* pada Gambar 3.5 menunjukkan *lag-lag* yang signifikan adalah *lag 1, lag 2, lag 3,* dan *lag 4*. Sehingga jaringan yang akan dibangun menggunakan 4 *input* yaitu y_{t-1}, y_{t-2}, y_{t-3} dan y_{t-4} dengan target y_t. Karena *input* data yang digunakan adalah data y_{t-1}, y_{t-2}, y_{t-3} dan y_{t-4}, maka data ke-1 sampai ke-4 tidak dapat digunakan, sehingga untuk

selanjutnya penelitian menggunakan data ke-5 sampai ke-92 yaitu sebanyak 88 data. Data-data tersebut dapat dilihat pada Lampiran 3.2.

Setelah *input* ditentukan, langkah selanjutnya adalah pembagian data. Pembagian data dimaksudkan untuk memisahkan data yang digunakan sebagai data *training* dan data *testing* jaringan. Dalam tugas akhir ini, akan digunakan perbandingan 75% untuk data *training* dan 25% untuk data *testing*. Dari 88 data yang terpakai didapatkan 66 data untuk data *training* dan 22 data untuk data *testing*. Data *training* dan data *testing* terlampir pada Lampiran 3.3.

Perancangan model jaringan perlu dilakukan untuk mendapatkan model jaringan terbaik. Sebelum perancangan model, perlu dilakukan normalisasi data agar data berdistribusi normal. Proses normalisasi menggunakan bantuan *Matlab R2009a*. Setelah data berdistribusi normal, dilakukan perancangan model terbaik. Model terbaik diperoleh dengan *trial* dan *error* terhadap beberapa macam arsitektur model. Pemilihan model terbaik dengan melihat nilai MSE dan MAPE hasil pembelajaran. Prosedur penentuan model terbaik adalah sebagai berikut:

a. Penentuan Jumlah Neuron Tersembunyi Jaringan

Jumlah neuron tersembunyi jaringan model diperoleh dengan cara *trial* dan *error* terhadap beberapa macam jaringan. Model dengan jumlah neuron terbaik dapat dilihat dari nilai *MSE* dan *MAPE* hasil pembelajaran tersebut. Jika nilai *MSE* dan *MAPE*-nya lebih kecil maka model tersebut lebih baik digunakan. Proses pembelajaran *recurrent neural network* algoritma *backpropagation* yang dimulai dari jaringan dengan 1 neuron pada lapisan tersembunyi hingga 20 neuron pada lapisan tersembunyi menggunakan *Matlab R2009a* terlihat pada Tabel 3.1

Tabel 3.1. Nilai MSE dan MAPE Hasil Pembelajaran

Jumlah Neuron	*Training*		*Testing*		Jumlah Neuron	*Training*		*Testing*	
	MSE (ribu)	MAPE	MSE (ribu)	MAPE		MSE (ribu)	MAPE	MSE (ribu)	MAPE
1	238000	3.43	1100000	10.31	11*	29500	1.26	2610000	3.79
2	183000	3	300000	9.01	12	56100	1.84	704000	5.53
3	143000	2.77	581000	8.31	13	38200	1.41	3010000	4.24
4	100000	2.36	622000	7.1	14	28700	1.23	1900000	3.68
5	102000	2.43	816000	7.31	15	32400	1.31	777000	3.94
6	69300	1.87	943000	5.61	16	48800	1.58	1370000	4.74
7	64600	1.8	1420000	5.39	17	46600	1.53	818000	4.59
8	39900	1.48	1710000	4.45	18	51000	1.67	1410000	5.01
9	63900	1.82	1630000	5.47	19	51200	1.62	1200000	4.87
10	49100	1.6211	1070000	4.86	20	52200	1.66	1770000	4.97

Keterangan: *) Model terbaik

Pada Tabel 3.1 terlihat bahwa model terbaik adalah model dengan jumlah neuron 11. Hal ini didasarkan dari nilai MSE dan MAPE menggunakan prinsip parsimoni. Prinsip parsimoni adalah memilih model yang memunyai MSE dan MAPE kecil dengan jaringan yang sederhana. Hal ini berarti, jika terdapat 2 model yang menghasilkan nilai MAPE dan MSE yang hampir sama, model yang dipilih adalah model dengan arsitektur yang paling sederhana.

b. Eliminasi *input* jaringan

Model terbaik adalah model dengan jaringan yang paling sederhana. Kesederhanaan model didasarkan pada jumlah neuron tersembunyi dan jumlah *input* jaringan. Dari maksud tersebut, setelah didapatkan jumlah neuron untuk model, akan dilakukan eliminasi *input* untuk mendapatkan model yang lebih sederhana. Eliminasi *input* dilakukan dengan cara melakukan pembelajaran dengan mengeliminasi satu *input* jaringan (dengan jumlah neuron tersembunyi 11) dan membandingkan nilai *MSE* dan *MAPE* jaringan. Tabel 3.2 berikut adalah nilai *MSE* dan *MAPE* hasil eliminasi *input*.

Tabel 3.2. Nilai MSE dan MAPE Hasil eliminasi

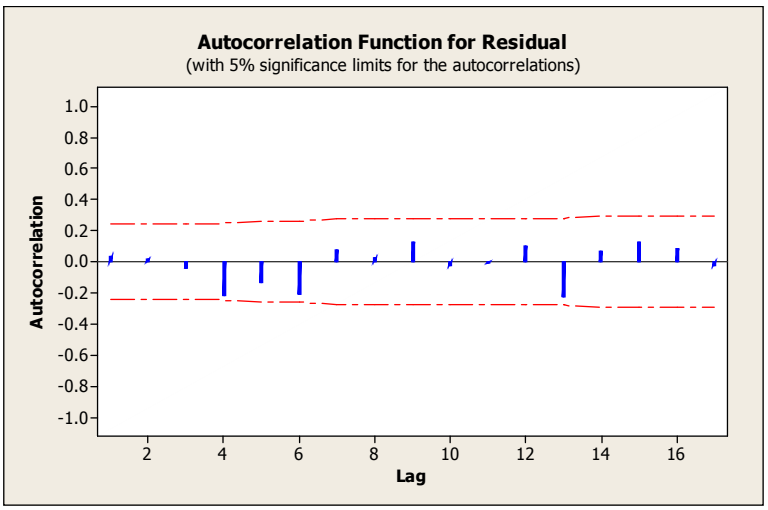
Eliminasi	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
y_{t-1}	7.04E+04	1.9365	1.03E+06	5.8096
y_{t-2}	1.11E+05	2.5071	4.99E+05	7.5213
y_{t-3}	1.08E+05	2.2781	1.74E+06	6.8344
y_{t-4}	8.04E+04	2.025	5.63E+05	6.0749
-*	2.95E+04*	1.2617*	2.61E+06*	3.785*

Keterangan: * model terbaik

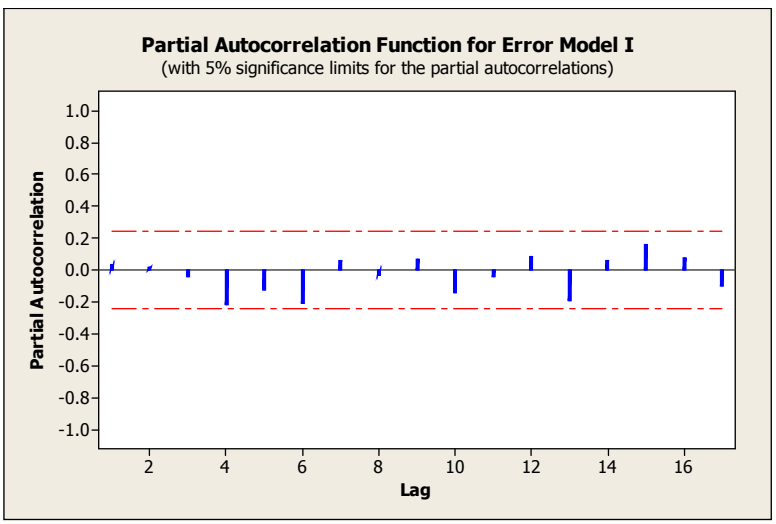
Berdasarkan nilai MSE dan MAPE hasil eliminasi yang tertulis pada Tabel 3.2, didapatkan model terbaik yaitu model dengan jaringan yang memiliki 4 *input* (tanpa eliminasi *input*).

Dari hasil pembelajaran model terbaik, diperoleh *output* jaringan. *Output* tersebut masih dalam bentuk normal. Sehingga untuk membawa *output* tersebut ke dalam bentuk data semula, digunakan fungsi *poststd* pada *Matlab*. Nilai target, *output* dan *error* dari hasil pembelajaran data *training* pada jaringan Elman telah terlampirkan pada Lampiran 3.4.

Evaluasi terhadap struktur *neural network* yang telah dibangun melalui pengecekan apakah *error* pada model tersebut *white noise* atau tidak. Evaluasi model dapat dilihat dari plot ACF dan PACF data *training* yang digunakan untuk melihat *error* yang bersifat *white noise* atau tidak. Jika *error white noise*, maka struktur jaringan yang terbentuk cocok digunakan untuk peramalan. Gambar 3.6 dan 3.7 adalah plot ACF dan PACF *error* data *training* hasil pembelajaran jaringan *recurrent neural network*.



Gambar 3.6. Plot ACF *Error* Model RNN



Gambar 3.7. Plot PACF *Error* Model RNN

Plot ACF dan PACF pada Gambar 3.6 dan 3.7 menunjukkan bahwa semua *lag* tidak ada yang melebihi selang kepercayaan. Dengan kata lain, semua *lag* tidak ada yang berbeda secara signifikan dari nol, artinya *error white noise*. Oleh karena itu, untuk hasil pembelajaran dengan menggunakan model *recurrent neural network* layak digunakan dalam peramalan.

2. Aplikasi *Recurrent Neuro Fuzzy* untuk Peramalan Penumpang Kereta Api Jabodetabek

Langkah awal peramalan menggunakan model *recurrent neuro fuzzy* adalah penentuan variabel *input. Input* model adalah *lag-lag* yang signifikan pada plot ACF data. Karena langkah penentuan *input* model *neuro fuzzy* sama dengan penentuan *input recurrent neural network*, maka *input* model *neuro fuzzy* adalah *input* hasil pembelajaran dari model *recurrent neural network.*

Input pada pembelajaran model *recurrent neural network* adalah data-data y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4} dengan *output* adalah data y_t. *Input* ini didasarkan dari hasil plot ACF pada Gambar 3.5. Setelah *input* diperoleh, akan dilakukan seleksi variabel *input* tersebut. Seleksi dilakukan dengan cara *trial* dan *error* kemudian melihat nilai MAPE dan MSE dari model hasil seleksi. Nilai MAPE dan MSE model yang dihasilkan selama proses seleksi *input* dapat dilihat pada Tabel 3.2. Dari Tabel 3.2 nilai MAPE dan MSE terkecil adalah model dengan *input* lengkap (tanpa eliminasi *input*). Sehingga akan dilakukan peramalan dengan model *recurrent neuro fuzzy* dengan *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4}.

Setelah mendapatkan jumlah neuron tersembunyi dan *input* jaringan, pada model *recurrent neuro fuzzy*, data pembelajaran harus dikelompokkan ke dalam kelas (*cluster*). Pengelompokan (*clustering*) data pembelajaran menggunakan menggunakan metode *Fuzzy C-Means.* Prosedur pengklasteran menggunakan FCM telah terbahas pada Bab II. Model akan memiliki *r* buah aturan R^s dengan *s=1,2,3,..r.* Pasangan *input-output* direpresentasikan sebagai (x_i^s, y_i^s), dengan *i=1,2,3……N_s*, N_s adalah banyak data yang masuk dalam kelas ke-*s.* Banyak *cluster* data pembelajaran minimal 2 *cluster*, dalam peramalan jumlah penumpang kereta api Jabodetabek ini data pembelajaran dibagi menjadi 3 *cluster*. Pengklasteran menggunakan FCM diperoleh 3 buah aturan R^s, *s=1,2,3*. Secara umum, aturan tersebut dapat dilihat pada Persamaan 2.49. Dari langkah ini akan diperoleh hasil pengklasteran data. Secara lengkap, pembagian *cluster* dapat dilihat pada Lampiran 3.5. Matriks pusat *cluster* pada data *training* dan data *testing* adalah sebagai berikut

Matriks pusat kelas *Training* adalah

$$V = \begin{vmatrix} 10565{,}92 & 10591{,}97 & 10663{,}33 & 10626{,}60 & 10616{,}90 \\ 10296{,}49 & 10341{,}70 & 10280{,}48 & 10312{,}00 & 10301{,}63 \\ 8698{,}70 & 8692{,}80 & 8741{,}28 & 8744{,}43 & 8849{,}17 \end{vmatrix}$$

Matriks pusat kelas *Testing* adalah

$$V = \begin{vmatrix} 10188{,}05 & 10384{,}99 & 10897{,}30 & 11326{,}73 & 11592{,}44 \\ 10224{,}72 & 9992{,}01 & 9954{,}81 & 9983{,}86 & 10116{,}90 \\ 11569{,}71 & 11737{,}61 & 11722{,}86 & 11651{,}33 & 11498{,}00 \end{vmatrix}$$

Matriks pusat *cluster* berukuran 3×5 sesuai dengan jumlah kelas yaitu 3 dan jumlah variabel *input* ditambah variabel *output* yaitu 5.

Bersama dengan pembagian *cluster* dari data, dilakukan pembelajaran *neural network* yang berhubungan dengan bagian anteseden (bagian *IF*) pada aturan-aturan inferensi *fuzzy*. Pembelajaran menggunakan *recurrent neural network* jaringan Elman menggunakan algoritma *bacpropagation*. Hasil pembelajaran adalah nilai keanggotaan data pada setiap *cluster*. Nilai keanggotaan tersebut yang selanjutnya digunakan untuk penentuan *output* pada inferensi *fuzzy* bagian antesenden. Tabel 3.3 adalah nilai keanggotaan dan hasil pengklasteran data *training* dan Tabel 3.4 adalah nilai keanggotaan dan hasil pengklasteran data *testing*.

Tabel 3.3. Nilai Keanggotaan dan Hasil Pengklasteran Data *Training*

Data ke	Nilai Keanggotaan			Maksimal	Kelas
	μ_i^1	μ_i^2	μ_i^3	μ_i^s	μ_i^s
1	0,011498	-0,03542	1,000517	1,000517	3
2	0,015768	-0,02039	1,004621	1,004621	3
…	…	…	…	…	…
63	0,778993	0,222602	-0,00302	0,778993	1
64	0,174029	0,833047	-0,00876	0,833047	2
65	-0,11955	1,11363	0,020654	1,11363	2
66	0,030473	0,971292	0,015968	0,971292	2

Tabel 3.4. Nilai Keanggotaan dan Hasil Pengklasteran Data *Testing*

Data ke	Nilai Keanggotaan			Maksimal	Kelas
	μ_i^1	μ_i^2	μ_i^3	μ_i^s	μ_i^s
67	-0.01617	0.32016	0.717576	0.717576	2
68	-0.03328	0.091155	0.966352	0.966352	2
…	…	…	…	…	…
86	1.412945	-0.40747	-0.02282	1.412945	1
87	1.285908	-0.28179	0.011517	1.285908	3
88	1.256853	-0.24807	-0.00877	1.256853	3

Nilai keanggotaan secara lengkap dapat dilihat pada Lampiran 3.6. Kecenderungan suatu pasangan data masuk ke suatu aturan inferensi $fuzzy\ A$ ke-*s* ditentukan oleh nilai keanggotaan masing-masing pasangan data pada tiap *cluster*. Jika suatu pasangan *training* masuk *cluster* ke-*s*, maka pasangan *training* tersebut akan masuk ke aturan inferensi $fuzzy\ A$ ke-*s* .

Berdasarkan Tabel 3.3, nilai keanggotaan maksimum dari pasangan *training* data pertama di aturan inferensi $fuzzy\ A$ ke-3, maka pasangan *training* pertama masuk *cluster* ke-3 dan akan mengalami pembelajaran di aturan *fuzzy* inferensi *A* ke-3. Demikian pula dengan pasangan *training* yang lain.

Berdasarkan Tabel 3.4 data *testing* pertama adalah di aturan inferensi $fuzzy\ A$ ke-2, maka pasangan *testing* pertama masuk *cluster* ke-2. Namun demikian nilai keanggotaan maksimum dari pasangan *testing* pertama adalah di aturan inferensi $fuzzy\ A$ ke-3. Hal ini dapat terjadi pada *testing*, sebab data *testing* hanya digunakan untuk pengecekan jaringan yang telah dibangun menggunakan data *training*.

Nilai keanggotaan data *training* dan data *testing* sebelumnya berupa himpunan *fuzzy* konversi ke himpunan tegas. Setiap vektor *input* pada *training* dan *testing* ditentukan $m_i = (m_i^1, m_i^2, \dots, m_i^r)$, $i = 1,2,\dots,66$ untuk *training* dan $i = 1,2,\dots,22$ untuk testing sesuai dengan Persamaan (3.8) berikut.

$$m_i^k = \begin{cases} 1; & k = s \\ 0; & k \neq s \end{cases} \quad (3.8)$$

Menurut persamaan tersebut, data yang masuk dalam *cluster* ke-*s* akan memiliki nilai keanggotaan 1 untuk *cluster* ke-*s* tersebut dan bernilai 0 untuk kelas yang lain. Nilai keanggotaan data *training* dan data *testing* sebelumnya berupa himpunan *fuzzy* konversi ke himpunan tegas yang dapat dilihat pada Lampiran 3.7 dan Lampiran 3.8.

Setelah didapatkan nilai keanggotaan bagian antesenden pada inferensi *fuzzy*, prosedur pemodelan selanjutnya mengikuti langkah-langkah sebagai berikut:

1. Pembelajaran *neural network* yang berhubungan dengan bagian konsekuen (bagian *THEN*) pada aturan-aturan inferensi *fuzzy*.

   Pembelajaran *neural network* pada bagian *THEN* dari R^s dengan *input* $x = \{y^s_{(t-4)}, y^s_{(t-3)}, y^s_{(t-2)}, y^s_{(t-1)}\}$ dan target *output* $y^s_{(t)}, s = 1,2,3; t = 1,2,..N_s$ dilakukan secara terpisah. Proses pembelajaran dibagi menjadi 3 proses pembelajaran sesuai dengan jumlah *cluster* yang ditentukan, yaitu $R^1(NN_1)$, $R^2(NN_2)$, dan $R^3(NN_3)$. Proses pembelajaran tiap NN_s menggunakan jaringan *recurrent neural network* algoritma *backpropagation*. Parameter yang digunakan: maksimum epoh = 5000, Kinerja tujuan= e^{-5} , *learning rate* = 0,1, momentum= 0,8, dan jumlah *epoch* = 500. Pembelajaran pada tiap jaringan syaraf NN_s dengan 3 aturan inferensi *fuzzy* adalah sebagai berikut.

a. Pembelajaran pada NN_1

   Proses pembelajaran jaringan dilakukan dengan *input* y_{t-4}, y_{t-3}, y_{t-2}, dan y_{t-1} dan target *output* y_t. Format aturan inferensi mengikuti aturan inferensi Soegeno yaitu:

   $IF\ x = (y_{(t-4)}, y_{(t-3)}, y_{(t-2)}, y_{(t-1)}) is\ A_s\ THEN\ y^s_{(t)} = f_s; s = 1,2,3$

   Sehingga untuk pembelajaran pada *NN1*, format untuk aturan inferensi pertama (R^1) adalah:

   $$R^1: IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1} is\ A_1\ THEN\ y_1 = f_1 \quad (3.9)$$

Secara lebih khusus dituliskan

$$R_1^1 \rightarrow if\ y_{t-4}\ is\ A_{11}, y_{t-3}\ is\ A_{12}\ , y_{t-2}\ is\ A_{13}, AND\ y_{t-1}\ is\ A_{14}\ THEN\ y_1 = f_1 \quad (3.10)$$

$$R_2^1 \rightarrow if\ y_{t-4}\ is\ A_{11}, y_{t-3}\ is\ A_{12}\ , y_{t-2}\ is\ A_{13}, AND\ y_{t-1}\ is\ A_{14}\ THEN\ y_1 = f_1 \quad (3.11)$$

$$R_3^1 \rightarrow if\ y_{t-4}\ is\ A_{11}, y_{t-3}\ is\ A_{12}\ , y_{t-2}\ is\ A_{13}, AND\ y_{t-1}\ is\ A_{14}\ THEN\ y_1 = f_1 \quad (3.12)$$

Data *training* dan *testing* yang akan digunakan untuk pembelajaran pada $R^1(NN_1)$ dapat dilihat pada Lampiran 3.9 dan Lampiran 3.10. Tabel 3.5 adalah tabel nilai MSE dan MAPE hasil pembelajaran

Tabel 3.5. Tabel Hasil Pembelajaran NN_1

Jumlah Neuron	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
1	1.14E+05	2.3414	9.30E+04	8.195
2	4.57E+04	1.2584	5.21E+06	4.4044
3	2.25E+04	0.8086	1.27E+06	2.83
4	5.79E+03	0.4024	3.36E+05	1.4083
5*	1.9573*	0.0082*	2.04E+06*	0.0288*
6	892.2721	0.1839	1.28E+05	0.6436
7	1.934	0.0096	3.22E+05	0.0334
8	1.9378	0.0075	7.42E+05	0.0261
9	1.9279	0.0076	1.61E+05	0.0265
10	1.9112	0.0085	2.01E+06	0.0298
11	1.9533	0.0076	4.47E+05	0.0266

Berdasarkan Tabel 3.5, model terbaik adalah model dengan jumlah neuron 5. Hal ini didasarkan oleh prinsip parsimoni yaitu model dengan nilai MSE dan MAPE kecil dengan jaringan sederhana. Oleh karena itu, jumlah neuron tersembunyi pada jaringan NN_1 adalah 5.

b. Pembelajaran pada NN_2

Proses pembelajaran jaringan dilakukan dengan *input* y_{t-4}, y_{t-3}, y_{t-2}, dan y_{t-1} dan target *output* y_t. Format aturan inferensi mengikuti aturan inferensi Soegeno yaitu:

$$IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1} is\ A_s\ THEN\ y_s = f_s; s = 1,2,3$$

Sehingga untuk pembelajaran pada NN_2, format untuk aturan *fuzzy* kedua (R^2) adalah:

$$R^2: IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1}\ is\ A_2\ THEN\ y_2 = f_2 \quad (3.13)$$

Secara lebih khusus dituliskan dengan

$$R_1^2 \rightarrow if\ y_{t-4}\ is\ A_{21}, y_{t-3}\ is\ A_{22}, y_{t-2}\ is\ A_{23}, AND\ y_{t-2} is\ A_{24} THEN\ y_2 = f_2 \quad (3.14)$$

$$R_2^2 \rightarrow if\ y_{t-4}\ is\ A_{21}, y_{t-3}\ is\ A_{22}, y_{t-2}\ is\ A_{23}, AND\ y_{t-2} is\ A_{24} THEN\ y_2 = f_2 \quad (3.15)$$

$$R_3^2 \rightarrow if\ y_{t-4}\ is\ A_{21}, y_{t-3}\ is\ A_{22}, y_{t-2}\ is\ A_{23}, AND\ y_{t-2} is\ A_{24} THEN\ y_2 = f_2 \quad (3.16)$$

Data *training* dan *testing* yang akan digunakan untuk pembelajaran pada $R^2(NN_2)$ dapat dilihat pada Lampiran 3.11 dan Lampiran 3.12. Tabel 3.6 menunjukkan nilai MSE dan MAPE hasil pembelajaran

Tabel 3.6. Tabel Hasil Pembelajaran NN_2

Jumlah Neuron	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
1	1.03E+05	2.4168	5.54E+05	10.7028
2	4.01E+04	1.3821	2.63E+06	6.1209
3	4.53E+04	1.4107	6.75E+05	6.2473
4	1.69E+04	0.7678	8.87E+05	3.4004
5	1.18E+04	0.5927	9.14E+05	2.6247
6	28.616	0.0299	1.37E+06	0.1325
7	1.56E+03	0.1679	1.02E+06	0.7435
8	412.2762	0.1154	6.70E+05	0.5111
9*	1.7418*	0.0085*	8.73E+05*	0.0377*
10	1.7659	0.008	1.34E+06	0.0353

Berdasarkan tabel 3.6, menggunakan prinsip parsimoni, model terbaik adalah model dengan jumlah neuron tersembunyi 9. Oleh karena itu, jumlah neuron tersembunyi pada jaringan NN_2 adalah 9.

c. Pembelajaran pada NN_3

Proses pembelajaran jaringan dilakukan dengan *input* y_{t-4}, y_{t-3}, y_{t-2}, dan y_{t-1} dan target *output* y_t. Format aturan inferensi mengikuti aturan inferensi Soegeno yaitu:

$$IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1} is\ A_s\ THEN\ y_s = f_s; s = 1,2,3$$

Sehingga untuk pembelajaran pada *NN3*, format untuk aturan *fuzzy* ketiga (R^3) adalah:

$$R^3: IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1}\ is\ A_3\ THEN\ y_3 = f_3 \quad (3.17)$$

Secara khusus dituliskan

$$R_1^3 \rightarrow if\ y_{t-4}\ is\ A_{31}, y_{t-3}\ is\ A_{32}, y_{t-2}\ is\ A_{33}, AND\ y_{t-1} is\ A_{34} THEN\ y_3 = f_3 \quad (3.18)$$

$$R_2^3 \rightarrow if\ y_{t-4}\ is\ A_{31}, y_{t-3}\ is\ A_{32}, y_{t-2}\ is\ A_{33}, AND\ y_{t-1} is\ A_{34} THEN\ y_3 = f_3 \quad (3.19)$$

$$R_3^3 \rightarrow if\ y_{t-4}\ is\ A_{31}, y_{t-3}\ is\ A_{32}, y_{t-2}\ is\ A_{33}, AND\ y_{t-1} is\ A_{34} THEN\ y_3 = f_3 \quad (3.20)$$

Data *training* dan *testing* yang akan digunakan untuk pembelajaran pada $R^3(NN_3)$ dapat dilihat pada Lampiran 3.13 dan Lampiran 3.14. Tabel 3.7 menunjukkan nilai MSE dan MAPE hasil pembelajaran

Tabel 3.7. Tabel Hasil Pembelajaran NN_3

Jumlah Neuron	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
1	9.36E+04	1.9483	4.11E+06	3.0307
2	8.06E+03	0.6983	5.21E+06	1.0862
3	1.23E+03	0.2863	1.35E+07	0.4453
4	37.5512	0.0438	2.55E+07	0.0681
5	81.4588	0.0677	7.23E+06	0.1053
6	30.6229	0.0392	2.91E+07	0.061
7*	6.3872*	0.019*	6.21E+06*	0.0296*
8	6.3111	0.0226	1.72E+07	0.0351
9	6.3812	0.0195	7.27E+06	0.0304
10	58.9594	0.0502	5.01E+06	0.0781

Berdasarkan Tabel 3.7 model terbaik adalah model dengan jumlah neuron tersembunyi adalah 7. Oleh karena itu, jumlah neuron tersembunyi pada jaringan NN_3 adalah 7.

2. Penyederhanaan bagian konsekuen (bagian *THEN*) menggunakan metode *backward*.

Setelah dilakukan pembelajaran menggunakan *recurrent neural network*, pada bagian ini akan dilakukan penyederhanaan bagian konsekuen. Pembelajaran *recurrent neural network* untuk penyederhanaan pada bagian *THEN* dari R^s menggunakan metode *backward* untuk jaringan *backpropagation*. Parameter yang digunakan antara lain maksimum epoh =

5000, Kinerja tujuan= e^{-5}, *learning rate* = 0,1, momentum= 0,8, dan jumlah *epoch* = 500.

a. Penyederhanaan pada NN_1

Setelah mendapatkan jumlah neuron tersembunyi jaringan, maka langkah selanjutnya adalah eliminasi *input* jaringan. *Input* jaringan akan dikeluarkan satu demi satu secara bergantian kemudian dipilih model yang menghasilkan MSE dan MAPE terkecil. Tabel 3.8 menunjukkan nilai MAPE dan MSE hasil eliminasi *input*

Tabel 3.8. Tabel Hasil Eliminasi *Input* NN_1

Eliminasi	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
-	1.9573	0.0082	2.04E+06	0.0288
y_{t-1}	6.96E+03	0.5483	4.23E+05	1.9191
y_{t-2}	364.6569	0.0966	1.85E+06	0.3382
y_{t-3}	443.5589	0.1	2.36E+06	0.35
y_{t-4}*	1.9493*	0.0102*	1.42E+06*	0.0357*

Berdasarkan Tabel 3.8, model terbaik adalah model dengan *input* jaringan tanpa y_{t-4} yaitu y_{t-1}, y_{t-2}, dan y_{t-3}. Oleh karena itu, jaringan pada NN_1 menggunakan jumlah neuron tersembunyi adalah 5 dengan *input* y_{t-1},y_{t-2}, dan y_{t-3}.

Model yang digunakan untuk peramalan adalah model Sogeno orde 1. Secara umum bentuk model *fuzzy* sugeno orde-1 adalah

$$IF\ (x_1 is\ A_1) \circ (x_2 is\ A_2) \circ \dots \circ (x_i is\ A_i)\ THEN\ y = p_1 \square x_1 + \cdots + p_i \square x_i + q$$

dengan $q, p_1, \dots p_i$ adalah koefisien persamaan linear. Koefisien konsekuen dari persamaan linear dicari dengan menggunakan metode LSE. Koefisien konsekuen yang diperoleh akan digunakan untuk memperbaiki bagian anteseden pada pembelajaran selanjutnya. Pembelajaran menggunakan jaringan *backpropagation*. Berikut koefisien yang diperoleh

Tabel 3.9 Tabel Koefisien Konsekuen NN_1

p0	*p1*	*p2*	*p3*
6.812	-0.1698	0.4853	0.6813

Aturan untuk aturan pertama R^1 adalah

$$R^1: IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1}\ is\ A_1\ THEN\ y_1 = f_1 \quad (3.21)$$

dengan

$$f_1 = -0{,}1698y_{t-3} + 0{,}4853y_{t-2} + 0{,}6813y_{t-1} + 6{,}812 \quad (3.22)$$

Hasil penyederhanaan bagian konsekuen pada pembelajaran *training* dan *testing* pada RNN_1 dapat dilihat pada Lampiran 3.15 dan Lampiran 3.16.

b. Pembelajaran pada NN_2

Jumlah neuron tersembunyi jaringan NN_2, adalah 9. Dengan jumlah neuron tersembunyi 9, dilakukan eliminasi *input*. Tabel 3.10 menunjukkan nilai MAPE dan MSE hasil eliminasi *input.*

Tabel 3.10. Tabel Hasil Eliminasi *Input* NN_2

	Training		*Testing*	
Eliminasi	MSE(ribu)	MAPE	MSE(ribu)	MAPE
-	1.7418	0.0085	8.73E+05	0.0377
y_{t-1}	8.0131	0.0198	1.46E+06	0.0876
y_{t-2}*	1.7674*	0.0092*	4.75E+05*	0.0406*
y_{t-3}	586.9364	0.1034	3.81E+06	0.4578
y_{t-4}	3.93E+03	0.3693	5.56E+05	1.6353

Berdasarkan Tabel 3.10, jaringan tanpa *input* y_{t-2} menghasilkan nilai MSE dan MAPE kecil. Oleh karena itu berdasarkan prinsip parsimoni, pembelajaran NN_2 menggunakan jumlah neuron tersembunyi adalah 9 dengan *input* $y_{t-1}, y_{t-3},$ dan y_{t-4}. Koefisien konsekuen hasil pembelajaran ditunjukkan pada Tabel 3.11 berikut:

Tabel 3.11. Tabel Konsekuen Parameter NN_2

p0	*p1*	*p2*	*p3*
9.26	0.1589	0.5029	0.3331

Aturan untuk aturan pertama R^2 adalah

$$R^2: IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1}\ is\ A_2\ THEN\ y_2 = f_2 \quad (3.23)$$

dengan,

$$f_2 = 0{,}1589y_{t-4} + 0{,}5029y_{t-3} + 0{,}3331y_{t-1} + 9{,}26 \quad (3.24)$$

Hasil penyederhanaan bagian konsekuen pada pembelajaran *training* dan *testing* pada RNN_2 dapat dilihat pada Lampiran 3.17 dan Lampiran 3.18.

c. Pembelajaran pada NN_3

Hasil pembelajaran NN_3 menghasilkan model terbaik dengan jumlah *input* adalah 7 . Tabel 3.12 menunjukkan nilai MAPE dan MSE hasil eliminasi *input.*

Tabel 3.12. Tabel Hasil Eliminasi *Input* NN_3

Eliminasi	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
-*	6.3872	0.019	6.21E+06	0.0296
y_{t-1}	67.3416	0.0664	9.23E+06	0.1033
y_{t-2}	13.1588	0.027	9.84E+06	0.042
y_{t-3}	6.36E+03	0.555	1.26E+07	0.8633
y_{t-4}	19.9545	0.0338	4.77E+06	0.0526

Berdasarkan Tabel 3,12, jaringan tanpa eliminasi *input* menghasilkan nilai MSE dan MAPE terkecil. Oleh karena itu, pembelajaran NN_3 menggunakan jumlah neuron tersembunyi adalah 7 dengan *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4}. Proses pencarian nilai koefisien konsekuen menggunakan *backpropagation*. Tabel 3.13 berikut adalah parameter konsekuen hasil pembelajaran

Tabel 3.13 Tabel Konsekuen Parameter NN_3

p0	*p1*	*p2*	*p3*	*p4*
2.1172	0.4711	-0.2356	-0.1739	0.9589

Aturan untuk aturan pertama R^3 adalah

$$R^3: IF\ x = y_{t-4}, y_{t-3}, y_{t-2}, y_{t-1}\ is\ A_3\ THEN\ y_3 = f_3 \quad (3.25)$$

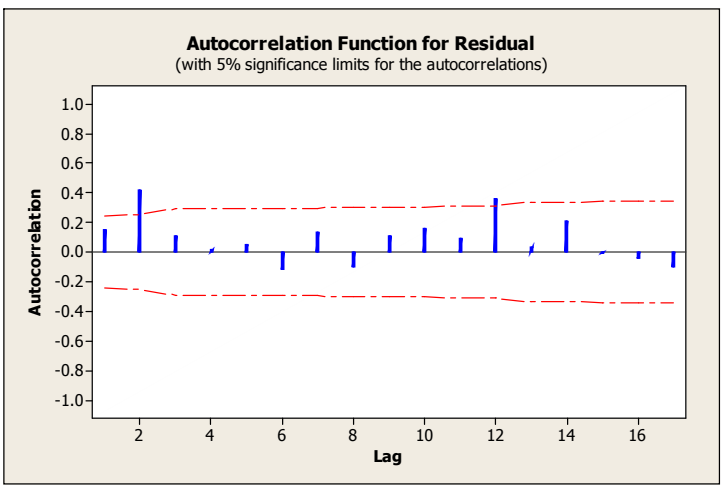
dengan,

$$f_3 = 0{,}4711y_{t-4} - 0{,}2356y_{t-3} - 0{,}1739y_{t-2} + 0{,}9589y_{t-1} + 2{,}1172 \quad (3.26)$$

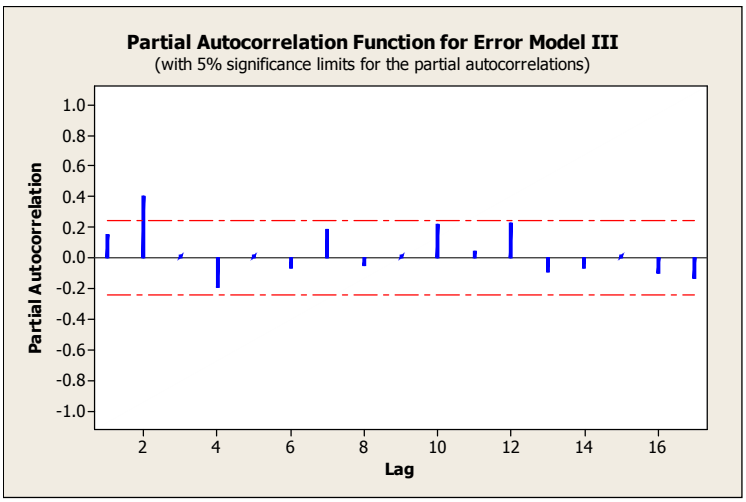
Hasil penyederhanaan bagian konsekuen pada pembelajaran *training* dan *testing* pada RNN_3 dapat dilihat pada Lampiran 3.19 dan Lampiran 3.20.

3. Uji kesesuaian model

Setelah model dibentuk, langkah selanjutnya adalah uji kesesuain model. Uji ini dilakukan untuk menguji apakah model layak digunakan untuk peramalan atau tidak. Nilai target, *output* dan *error* dari hasil pembelajaran data *training* pada jaringan *neuro fuzzy* telah terlampirkan pada Lampiran 3.21. Gambar 3.8 dan Gambar 3.9 adalah plot ACF dan PACF *error* data *training* hasil pembelajaran jaringan *recurrent neuro fuzzy*



Gambar 3.8. Plot ACF *Error* RNF



Gambar 3.9. Plot PACF *Error* RNF

Plot ACF yang terlihat pada Gambar 3.8 menunjukkan bahwa ada *lag* autokorelasi yang melebihi selang kepercayaan, yaitu *lag* 2 dan *lag* 12. Begitu juga dengan plot PACF pada Gambar 3.9 terdapat *lag* yang melebihi selang kepercayaan yaitu *lag* 2. Menurut kriteria signifikansi, ada *lag* ACF dan PACF yang berbeda secara signifikan dari nol. Dari hal tersebut dapat disimpulkan bahwa *error* tidak *white noise*. Oleh karena itu, untuk hasil pembelajaran dengan menggunakan jaringan *recurrent neuro fuzzy* kurang baik jika digunakan dalam peramalan.

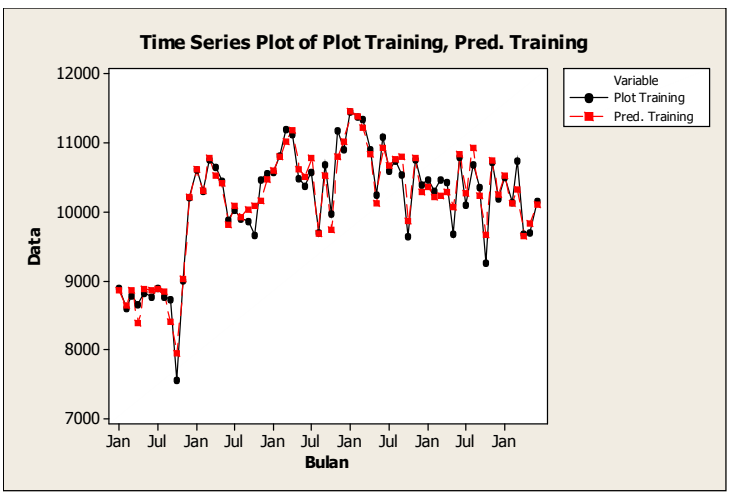
D. Peramalan Jumlah Penumpang Kereta Api Jabodetabek dengan Model Terpilih

Proses peramalan ini menggunakan model jaringan terbaik yang diperoleh dari pembelajaran data pada data *training* dan data *testing*. Pemilihan model terbaik dengan membandingkan nilai MSE dan MAPE keempat model. Keakuratan hasil pengujian dengan model I: model Elman RNN dengan *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4} dan model II: model *recurren neuro fuzzy* dengan *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4} dapat dilihat dari nilai MSE dan MAPE pada Tabel 3.14. Kedua model menggunakan 1 lapisan tersembunyi dengan 11 neuron dan 1 lapisan *input*.

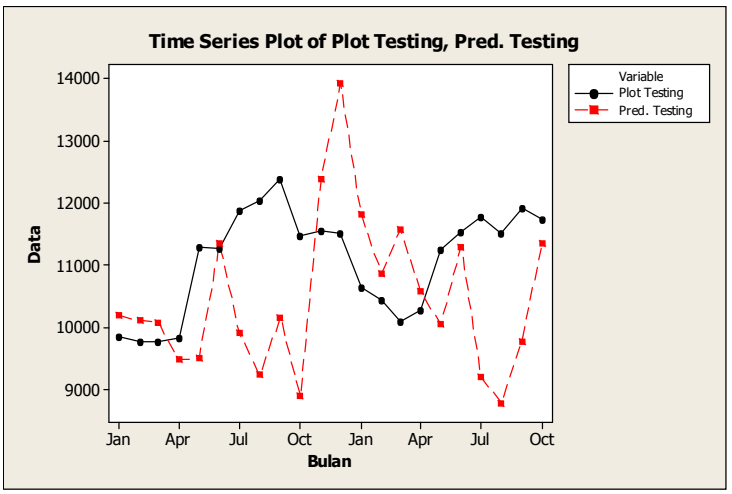
Tabel 3.14. Tabel MAPE dan MSE Model

Model	*Training*		*Testing*	
	MSE(ribu)	MAPE	MSE(ribu)	MAPE
RNN *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4}	29500	1.2617	2610000	3.785
Recurren Neuro Fuzzy input y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4}	38600	5.2468	2000000	15.7403

Peramalan dilakukan dengan bantuan program *Matlab* R2009a. Hasil *output* menggunakan model *recurrent neural network* dapat dilihat pada Lampiran 3.4. Gambar 3.10 dan Gambar 3.11 menunjukkan plot data aktual dan hasil peramalan dari penumpang kereta api Jabodetabek model *recurrent neural network*.



Gambar 3.10. Plot Peramalan dan Aktual Data *Training* Model RNN



Gambar 3.11. Plot Peramalan dan Aktual Data *Testing* Model RNN

Penentuan *output* akhir jaringan *recurrent neuro fuzzy* dapat diperoleh dengan persamaan sebagai berikut (Lin dan Lee, 1996:510):

$$\bar{y_i} = \frac{\sum_{s=1}^{r} \mu_{As}(x_i) f_s(x_i)}{\sum_{s=1}^{r} \mu_{As}(x_i)}; i = 1, 2, \dots, N \qquad (3.27)$$

dengan,

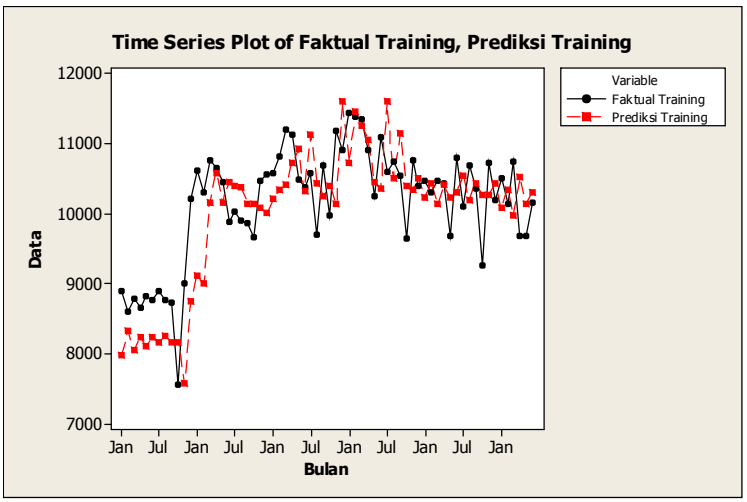
s : banyak aturan *inferensi fuzzy*, s=1, 2,....r

$\bar{y_i}$: *output* akhir jaringan.

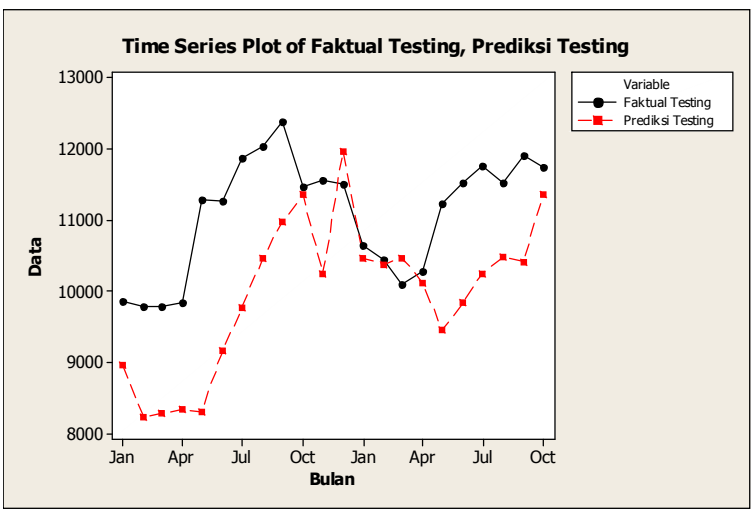
$\mu_{As}(x_i)$: nilai keanggotaan tiap x_i dalam himpunan *fuzzy A* pada aturan *fuzzy* ke-s bagian anteseden

$f_s(x_i)$: *output* jaringan hasil pembelajaran tiap x_i pada setiap aturan inferensi *fuzzy* R^s pada bagian konsekuen

Nilai $f_s(x_i)$ terlampir pada Lampiran 3.22 Contoh perhitungan *output* dapat dilihat pada Lampiran 3.23. Hasil penentuan *output* menggunakan model *recurrent neuro fuzzy* dapat dilihat pada Lampiran 3.21. Gambar 3.12 dan Gambar 3.13 adalah plot data aktual dan hasil peramalan menggunakan model *recurrent neuro fuzzy.*



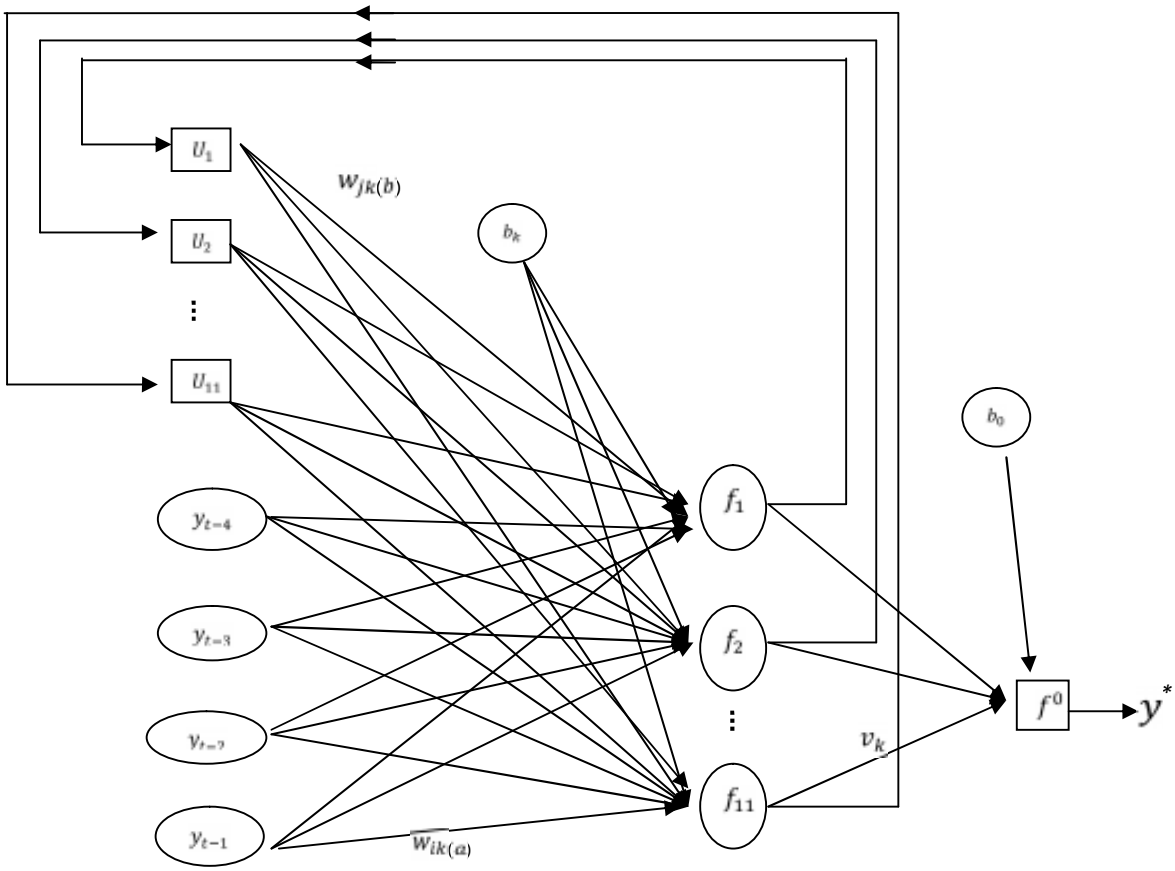
Gambar 3.12. Plot Data Aktual dan Peramalan *Training* RNF



Gambar 3.13. Plot Data Aktual dan Peramalan *Testing* RNF

Berdasarkan Tabel 3.14 terlihat bahwa MSE dan MAPE terkecil dihasilkan oleh model Elman RNN dengan *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4}. Dari hasil pengujian diperoleh nilai MSE dan MAPE pada data *training* berturut-turut sebesar 29500 dan 1,2617%. Nilai MSE dan MAPE hasil pengujian pada data *testing* adalah 2610000 dan 3,785%. Disamping itu, model dari hasil pengujian kesesuaian model, model *recurrent neuro fuzzy* kurang baik digunakan untuk peramalan karena tidak memenuhi kriteria *white noise*.

Model terbaik adalah model RRN, maka model terpilih yang akan digunakan peramalan adalah model RNN dengan *input* y_{t-1}, y_{t-2}, y_{t-3}, dan y_{t-4}. Gambar 3.14 adalah arsitektur jaringan Elman RNN dengan 4 *input*, 1 lapisan tersembunyi dengan 11 neuron, dan 1 lapisan *output*.



Gambar 3.14. Arsitektur Jaringan Model RNN

Persamaan matematis model tersebut adalah persamaan

$$y^* = \sum_{k=1}^{11} v_k \frac{1-\exp\left(-\left(\sum_{i=1}^{4}\sum_{k=1}^{11} y_{t-i} w_{ik(a)} + \sum_{j=1}^{11}\sum_{k=1}^{11} u_j w_{jk(b)} + b_k\right)\right)}{1+\exp\left(-\left(\sum_{i=1}^{4}\sum_{k=1}^{11} y_{t-i} w_{ik(a)} + \sum_{j=1}^{11}\sum_{k=1}^{11} u_j w_{jk(b)} + b_k\right)\right)} + b_0 \qquad (3.28)$$

Model RNN adalah model dengan koneksi umpan balik dari neuron tersembunyi menuju neuron tambahan pada lapisan *output*. Dari neuron tambahan pada lapisan *output* tersebut, diperoleh dua bobot tambahan yaitu bobot dari neuron tambahan menuju neuron tersembunyi serta bobot dari neuron tambahan menuju lapisan *output*. Operasi keluaran lapisan lambahan (*input*) ke-*k* ke lapisan tersembunyi dapat dihitung dengan persamaan

$$u_j = \frac{1-\exp\left(-\left(\sum_{i=1}^{4}\sum_{k=1}^{11} y_{t-i} w_{ik(a)} + b_k\right)\right)}{1+\exp\left(-\left(\sum_{i=1}^{4}\sum_{k=1}^{11} y_{t-i} w_{ik(a)} + b_k\right)\right)} \tag{3.29}$$

dan operasi keluaran dari neuron tambahan menuju ke lapisan *output* dapat dihitung dengan persamaan

$$T_k = \frac{1-\exp\left(-\left(\sum_{i=1}^{4}\sum_{k=1}^{11} y_{t-i} w_{ik(a)} + \sum_{j=1}^{11}\sum_{k=1}^{11} u_j w_{jk(b)} + b_k\right)\right)}{1+\exp\left(-\left(\sum_{i=1}^{4}\sum_{k=1}^{11} y_{t-i} w_{ik(a)} + \sum_{j=1}^{11}\sum_{k=1}^{11} u_j w_{jk(b)} + b_k\right)\right)} \tag{3.30}$$

Selanjutnya dari nilai-nilai yang didapatkan, dapat digunakan untuk menghitung *output* jaringan. *Output* jaringan dapat dihitung menggunakan persamaan

$$y^{\square} = \sum_{k=1}^{4} v_{k1} T_k + b_0. \tag{3.31}$$

Setelah mendapatkan model terbaik, akan diramalkan jumlah penumpang kereta api Jabodetabek menggunakan persamaan (3.28). Berdasarkan pelatihan menggunakan struktur jaringan Elman RNN dengan algoritma *backpropagation* dengan menggunakan program *matlab*, diperoleh bobot bobot sebagai berikut:

$b_k = [0{,}124;\ -2.488;\ 0{,}995, \ldots .; 1{,}603; 2{,}535;\ -1{,}690]$

$b_0 = -0{,}025$

$w_{jk(b)}$ =

[-0,476	-0,013	-0,082	⋯	0,211	-0,525	-0,444;
-0,002	0,582	-0,242	⋯	0,271	0,608	0,535;
0,072	-0,458	-0,531	⋯	0,583	-0,233	-0,353;
⋮	⋮	⋮	⋮	⋮	⋮	⋮
0,020	-0,124	-0,642	⋯	-0,612	0,626	0,518;
-0,014	-0,014	-0,246	⋯	-0,351	-0,149	-0,470]

$w_{ik(a)}$ =

[0,912	-1,272	-0,328	⋯	-0,900	0,737	-1,071;
1,007	2,936	1,789	⋯	2,582	0,942	-0,432;
-1,034	1,213	-1,663	⋯	2,659	-1,739	0,606;
1,358	-2,403	-0,918	⋯	-2,578	-2,866	1,169]

$v_k = [-2{,}483;\ 2{,}666; 1{,}688; \ldots.; -2{,}433; -2{,}094; -1{,}145]$

Bobot-bobot hasil pembelajaran secara lengkap dapat dilihat pada Lampiran 3.24. Selanjutnya bobot-bobot yang diperoleh disubtitusikan ke dalam persamaan model yaitu Persamaan (3.28):

$$\hat{y} = -2{,}483\frac{1 - e^{-((1{,}514)(0{,}912)+\cdots+(1{,}483)(1{,}359)+(0{,}911)(-0{,}476)+\cdots+(-0{,}507)(-0{,}014)+(0{,}124))}}{1 + e^{-((1{,}514)(0{,}912)+\cdots+(1{,}483)(1{,}359)+(0{,}911)(-0{,}476)+\cdots+(-0{,}507)(-0{,}014)+(0{,}124))}}$$
$$+ 2{,}666\frac{1 - e^{-((1{,}514)(-1{,}272)+\cdots+(1{,}483)(-2{,}403)+(0{,}911)(-0{,}013)+\cdots+(-0{,}507)(-0{,}014)+(-2{,}489))}}{1 + e^{-((1{,}514)(-1{,}272)+\cdots+(1{,}483)(-2{,}403)+(0{,}911)(-0{,}013)+\cdots+(-0{,}507)(-0{,}014)+(-2{,}489))}}$$
$$+ \cdots. -1{,}145\frac{1 - e^{-((1{,}514)(-1{,}071)+\cdots+(1{,}483)(1{,}169)+(0{,}911)(-0{,}444)+\cdots+(-0{,}507)(-0{,}47)+(-1{,}169))}}{1 + e^{-((1{,}514)(-1{,}071)+\cdots+(1{,}483)(1{,}169)+(0{,}911)(-0{,}444)+\cdots+(-0{,}507)(-0{,}47)+(-1{,}169))}}$$
$$- 0{,}025$$

Berdasarkan Persamaan (3.29) diperoleh:

Operasi keluaran dari lapisan *input* 1 ke lapisan tersembunyi

$$u\_net_1 = y_{t-4}w_{11(a)} + y_{t-3}w_{21(a)} + y_{t-2}w_{31(a)} + y_{t-1}w_{41(a)} + b_1$$
$$= (1{,}514)(0{,}912) + (1{,}255)(1{,}007) + (1{,}658)(-1{,}034)$$
$$+ (1{,}483)(1{,}359) + (0{,}124) = 3{,}069$$
$$u_1 = f(u\_net_1) = \frac{1 - e^{-(3{,}069)}}{1 + e^{-(3{,}069)}} = 0{,}9112647$$

Operasi keluaran dari lapisan *input* 2 ke lapisan tersembunyi

$$u\_net_2 = y_{t-4}w_{12(a)} + y_{t-3}w_{22(a)} + y_{t-2}w_{32(a)} + y_{t-1}w_{42(a)} + b_2$$
$$= (1{,}514)(-1{,}272) + (1{,}255)(2{,}936) + (1{,}658)(1{,}213)$$
$$+ (1{,}483)(-2{,}403) + (-2{,}489) = -2{,}282$$
$$u_2 = f(u\_net_2) = \frac{1 - e^{-(-2{,}282)}}{1 + e^{-(-2{,}282)}} = -0{,}8147$$

Operasi keluaran dari lapisan *input* 3 ke lapisan tersembunyi

$$u\_net_3 = y_{t-4}w_{13(a)} + y_{t-3}w_{23(a)} + y_{t-2}w_{33(a)} + y_{t-1}w_{43(a)} + b_3$$
$$= (1{,}514)(-0{,}328) + (1{,}255)(1{,}789) + (1{,}658)(-1{,}663)$$
$$+ (1{,}483)(-0{,}918) + (0{,}995) = -1{,}376$$
$$u_3 = f(u\_net_3) = \frac{1 - e^{-(-1{,}376)}}{1 + e^{-(-1{,}376)}} = -0{,}59688$$

Operasi keluaran dari lapisan *input* 4 ke lapisan tersembunyi

$$u\_net_4 = y_{t-4}w_{14(a)} + y_{t-3}w_{24(a)} + y_{t-2}w_{34(a)} + y_{t-1}w_{44(a)} + b_4$$
$$= (1{,}514)(-0{,}303) + (1{,}255)(-0{,}250) + (1{,}658)(0{,}522)$$
$$+ (1{,}483)(0{,}090) + (-0{,}236) = -0{,}009$$
$$u_4 = f(u\_net_4) = \frac{1 - e^{-(-0{,}009)}}{1 + e^{-(-0{,}009)}} = -0{,}00459$$

Operasi keluaran dari lapisan *input* 5 ke lapisan tersembunyi

$$u\_net_5 = y_{t-4}w_{15(a)} + y_{t-3}w_{25(a)} + y_{t-2}w_{35(a)} + y_{t-1}w_{45(a)} + b_5$$
$$= (1{,}514)(-0{,}464) + (1{,}255)(0{,}132) + (1{,}658)(-0{,}446)$$
$$+ (1{,}483)(-1{,}265) + (-0{,}871) = -4{,}022$$
$$u_5 = f(u\_net_5) = \frac{1 - e^{-(-4{,}022)}}{1 + e^{-(-4{,}022)}} = -0{,}96481$$

Operasi keluaran dari lapisan *input* 6 ke lapisan tersembunyi

$$u\_net_6 = y_{t-4}w_{16(a)} + y_{t-3}w_{26(a)} + y_{t-2}w_{36(a)} + y_{t-1}w_{46(a)} + b_6$$
$$= (1{,}514)(0{,}359) + (1{,}255)(-2{,}262) + (1{,}658)(-1{,}956)$$
$$+ (1{,}483)(1{,}932) + (-0{,}342) = -3{,}015$$
$$u_6 = f(u\_net_6) = \frac{1 - e^{-(-3{,}015)}}{1 + e^{-(-3{,}015)}} = -0{,}90652$$

Operasi keluaran dari lapisan *input* 7 ke lapisan tersembunyi

$$u\_net_7 = y_{t-4}w_{17(a)} + y_{t-3}w_{27(a)} + y_{t-2}w_{37(a)} + y_{t-1}w_{47(a)} + b_7$$
$$= (1{,}514)(-0{,}965) + (1{,}255)(1{,}185) + (1{,}658)(1{,}833)$$
$$+ (1{,}483)(-1{,}348) + (-1{,}189) = -0{,}123$$
$$u_7 = f(u\_net_7) = \frac{1 - e^{-(-0{,}123)}}{1 + e^{-(-0{,}123)}} = -0{,}0615$$

Operasi keluaran dari lapisan *input* 8 ke lapisan tersembunyi

$$u\_net_8 = y_{t-4}w_{18(a)} + y_{t-3}w_{28(a)} + y_{t-2}w_{38(a)} + y_{t-1}w_{48(a)} + b_8$$
$$= (1{,}514)(0{,}828) + (1{,}255)(-0{,}822) + (1{,}658)(-0{,}493)$$
$$+ (1{,}483)(1{,}526) + (0{,}697) = 2{,}364$$
$$u_8 = f(u\_net_8) = \frac{1 - e^{-(2{,}364)}}{1 + e^{-(2{,}364)}} = -0{,}8281$$

Operasi keluaran dari lapisan *input* 9 ke lapisan tersembunyi

$$u\_net_9 = y_{t-4}w_{19(a)} + y_{t-3}w_{29(a)} + y_{t-2}w_{39(a)} + y_{t-1}w_{49(a)} + b_9$$
$$= (1{,}514)(-0{,}900) + (1{,}255)(2{,}582) + (1{,}658)(2{,}659)$$
$$+ (1{,}483)(-2{,}578) + (1{,}603) = 4{,}068$$
$$u_9 = f(u\_net_9) = \frac{1 - e^{-(4{,}068)}}{1 + e^{-(4{,}068)}} = -0{,}966344$$

Operasi keluaran dari lapisan *input* 10 ke lapisan tersembunyi

$$u\_net_{10} = y_{t-4}w_{1\_10(a)} + y_{t-3}w_{2\_10(a)} + y_{t-2}w_{3\_10(a)} + y_{t-1}w_{4\_10(a)} + b_{10}$$
$$= (1{,}514)(0{,}737) + (1{,}255)(0{,}942) + (1{,}658)(-1{,}739)$$
$$+ (1{,}483)(-2{,}866) + (2{,}535) = -2{,}301$$
$$u_{10} = f(u\_net_{10}) = \frac{1 - e^{-(-2{,}301)}}{1 + e^{-(-2{,}301)}} = -0{,}81796$$

Operasi keluaran dari lapisan *input* 11 ke lapisan tersembunyi

$$u\_net_{11} = y_{t-4}w_{1\_11(a)} + y_{t-3}w_{2\_11(a)} + y_{t-2}w_{3\_11(a)} + y_{t-1}w_{4\_11(a)} + b_{11}$$
$$= (1{,}514)(-1{,}071) + (1{,}255)(-0{,}432) + (1{,}658)(0{,}606)$$
$$+ (1{,}483)(1{,}169) + (-1{,}169) = -1{,}117$$
$$u_{11} = f(u\_net_{11}) = \frac{1 - e^{-(-1{,}117)}}{1 + e^{-(-1{,}117)}} = -0{,}50698$$

Kemudian, berdasarkan Persamaan (3.30) diperoleh:

Operasi keluaran neuron 1 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_1 = y_{t-4}w_{11(a)} + y_{t-3}w_{21(a)} + y_{t-2}w_{31(a)} + y_{t-1}w_{41(a)} + b_1$$
$$+ u_1w_{11(b)} + u_2w_{21(b)} + u_3w_{31(b)} + u_{41}w_{41(b)} + u_5w_{51(b)}$$
$$+ u_6w_{61(b)} + u_7w_{71(b)} + u_8w_{81(b)} + u_9w_{91(b)} + u_{10}w_{10_{1(b)}}$$
$$+ u_{11}w_{11_{1(b)}}$$

yang dapat disederhanakan menjadi

$$T\_net_1 = u\_net_1 + u_1 w_{11(b)} + u_2 w_{21(b)} + u_3 w_{31(b)} + u_1 w_{41(b)} + u_5 w_{51(b)} + u_6 w_{61(b)} + u_7 w_{71(b)} + u_8 w_{81(b)} + u_9 w_{91(b)} + u_{10} w_{10\_1(b)} + u_{11} w_{11\_1(b)}$$

$$= 3{,}069 + (0{,}911)(-0{,}476) + (-0{,}815)(-0{,}002) + (-0{,}597)(0{,}072) + (-0{,}005)(-0{,}378) + (-0{,}965)(-0{,}182) + (-0{,}907)(-0{,}247) + (-0{,}062)(-0{,}075) + (0{,}828)(0{,}702) + (0{,}966)(-0{,}483) + (-0{,}818)(0{,}020) + (-0{,}507)(-0{,}014) = 3{,}11$$

$$T_1 = f(T\_net_1) = \frac{1 - e^{-(3{,}11)}}{1 + e^{-(3{,}11)}} = 0{,}914274$$

Operasi keluaran neuron 2 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_2 = u\_net_2 + u_1 w_{12(b)} + u_2 w_{22(b)} + u_3 w_{32(b)} + u_4 w_{42(b)} + u_5 w_{52(b)} + u_6 w_{62(b)} + u_7 w_{72(b)} + u_8 w_{82(b)} + u_9 w_{92(b)} + u_{10} w_{10\_2(b)} + u_{11} w_{11\_2(b)}$$

$$= -2{,}282 + (0{,}911)(-0{,}013) + (-0{,}815)(0{,}582) + \cdots + (-0{,}818)(-0{,}124) + (-0{,}507)(-0{,}014) = -2{,}86$$

$$T_2 = f(T\_net_2) = \frac{1 - e^{-(-2{,}86)}}{1 + e^{-(-2{,}86)}} = -0{,}89138$$

Operasi keluaran neuron 3 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_3 = u\_net_3 + u_1 w_{13(b)} + u_2 w_{23(b)} + u_3 w_{33(b)} + u_4 w_{43(b)} + u_5 w_{53(b)} + u_6 w_{63(b)} + u_7 w_{73(b)} + u_8 w_{83(b)} + u_9 w_{93(b)} + u_{10} w_{10\_3(b)} + u_{11} w_{11\_3(b)}$$
$$= -1{,}377 + (0{,}911)(-0{,}082) + (-0{,}815)(-0{,}241) + \cdots + (-0{,}818)(-0{,}642) + (-0{,}507)(-0{,}246) = -0{,}22$$

$$T_3 = f(T\_net_3) = \frac{1 - e^{-(-0{,}22)}}{1 + e^{-(-0{,}22)}} = -0{,}10927$$

Operasi keluaran neuron 4 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_4 = u\_net_4 + u_1 w_{14(b)} + u_2 w_{24(b)} + u_3 w_{34(b)} + u_4 w_{44(b)} + u_5 w_{54(b)} + u_6 w_{64(b)} + u_7 w_{74(b)} + u_8 w_{84(b)} + u_9 w_{94(b)} + u_{10} w_{10\_4(b)} + u_{11} w_{11\_4(b)}$$
$$= -0{,}009 + (0{,}911)(0{,}208) + (-0{,}815)(0{,}121) + \cdots + (-0{,}818)(-0{,}369) + (-0{,}507)(0{,}567) = 0{,}99$$

$$T_4 = f(T_{net_4}) = \frac{1 - e^{-(0{,}99)}}{1 + e^{-(0{,}99)}} = 0{,}459773$$

Operasi keluaran neuron 5 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_5 = u\_net_5 + u_1 w_{15(b)} + u_2 w_{25(b)} + u_3 w_{35(b)} + u_4 w_{45(b)} + u_5 w_{55(b)} + u_6 w_{65(b)} + u_7 w_{75(b)} + u_8 w_{85(b)} + u_9 w_{95(b)} + u_{10} w_{10\_5(b)} + u_{11} w_{11\_5(b)}$$

$$= -4{,}022 + (0{,}911)(0{,}322) + (-0{,}815)(-0{,}425) + \cdots + (-0{,}818)(-0{,}58) + (-0{,}507)(-0{,}201) = -2{,}15$$

$$T_5 = f(T\_net_5) = \frac{1 - e^{-(-2{,}15)}}{1 + e^{-(-2{,}15)}} = -0{,}79211$$

Operasi keluaran neuron 6 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_6 = u\_net_6 + u_1 w_{16(b)} + u_2 w_{26(b)} + u_3 w_{36(b)} + u_4 w_{46(b)} + u_5 w_{56(b)} + u_6 w_{66(b)} + u_7 w_{76(b)} + u_8 w_{86(b)} + u_9 w_{96(b)} + u_{10} w_{10\_6(b)} + u_{11} w_{11\_6(b)}$$

$$= -3{,}015 + (0{,}911)(0{,}366) + (-0{,}815)(0{,}361) + \cdots + (-0{,}818)(-0{,}455) + (-0{,}507)(-0{,}559) = -2{,}47$$

$$T_6 = f(T\_net_6) = \frac{1 - e^{-(-2{,}47)}}{1 + e^{-(-2{,}47)}} = -0{,}84473$$

Operasi keluaran neuron 7 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_7 = u\_net_7 + u_1 w_{17(b)} + u_2 w_{27(b)} + u_3 w_{37(b)} + u_4 w_{47(b)} + u_5 w_{57(b)} + u_6 w_{67(b)} + u_7 w_{77(b)} + u_8 w_{87(b)} + u_9 w_{97(b)} + u_{10} w_{10\_7(b)} + u_{11} w_{11\_7(b)}$$

$$= -0{,}123 + (0{,}911)(-0{,}362) + (-0{,}815)(-0{,}396) + \cdots + (-0{,}818)(-0{,}42) + (-0{,}507)(0{,}453) = -1{,}21$$

$$T_7 = f(T\_net_7) = \frac{1 - e^{-(-1{,}21)}}{1 + e^{-(-1{,}21)}} = -0{,}53912$$

Operasi keluaran neuron 8 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_8 = u\_net_8 + u_1 w_{18(b)} + u_2 w_{28(b)} + u_3 w_{38(b)} + u_4 w_{48(b)} + u_5 w_{58(b)} + u_6 w_{68(b)} + u_7 w_{78(b)} + u_8 w_{88(b)} + u_9 w_{98(b)} + u_{10} w_{10\_8(b)} + u_{11} w_{11\_8(b)}$$

$$= 2{,}364 + (0{,}911)(0{,}28) + (-0{,}815)(0{,}009) + \cdots + (-0{,}818)(-0{,}129) + (-0{,}507)(-0{,}172) = 3{,}04$$

$$T_8 = f(T\_net_8) = \frac{1 - e^{-(3{,}04)}}{1 + e^{-(3{,}04)}} = 0{,}908562$$

Operasi keluaran neuron 9 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_9 = u\_net_9 + u_1 w_{19(b)} + u_2 w_{29(b)} + u_3 w_{39(b)} + u_4 w_{49(b)} + u_5 w_{59(b)} + u_6 w_{69(b)} + u_7 w_{79(b)} + u_8 w_{89(b)} + u_9 w_{99(b)} + u_{10} w_{10\_9(b)} + u_{11} w_{11\_9(b)}$$

$$= 4{,}068 + (0{,}911)(0{,}211) + (-0{,}815)(0{,}271) + \cdots + (-0{,}818)(-0{,}612) + (-0{,}507)(-0{,}351) = 4{,}93$$

$$T_9 = f(T\_net_9) = \frac{1 - e^{-(4{,}93)}}{1 + e^{-(4{,}93)}} = 0{,}985621$$

Operasi keluaran neuron 10 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_{10} = u\_net_{10} + u_1 w_{1\_10(b)} + u_2 w_{2\_10(b)} + u_3 w_{3\_10(b)} + u_4 w_{4\_10(b)} + u_5 w_{5\_10(b)} + u_6 w_{6\_10(b)} + u_7 w_{7\_10(b)} + u_8 w_{8\_10(b)} + u_9 w_{9\_10(b)} + u_{10} w_{10\_10(b)} + u_{11} w_{11\_10(b)}$$

$$= -2{,}301 + (0{,}911)(-0{,}525) + (-0{,}815)(0{,}608) + \cdots + (-0{,}818)(0{,}626) + (-0{,}507)(-0{,}149) = -3{,}694$$

$$T_{10} = f(T\_net_{10}) = \frac{1 - e^{-(-3{,}694)}}{1 + e^{-(-3{,}694)}} = -0{,}95146$$

Operasi keluaran neuron 11 pada lapisan tersembunyi dengan neuron tambahan menuju ke lapisan *output*:

$$T\_net_{11} = u\_net_{11} + u_1 w_{1\_11(b)} + u_2 w_{2\_11(b)} + u_3 w_{3\_11(b)} + u_4 w_{4\_11(b)} + u_5 w_{5\_11(b)} + u_6 w_{6\_11(b)} + u_7 w_{7\_11(b)} + u_8 w_{8\_11(b)} + u_9 w_{9\_11(b)} + u_{10} w_{10\_11_{(b)}} + u_{11} w_{11\_11_{(b)}}$$

$$= -1{,}117 + (0{,}911)(-0{,}444) + (-0{,}815)(0{,}535) + \cdots + (-0{,}818)(0{,}518) + (-0{,}507)(-0{,}47) = -1{,}39$$

$$T_{11} = f(T\_net_{11}) = \frac{1 - e^{-(-1{,}39)}}{1 + e^{-(-1{,}39)}} = -0{,}6015$$

Selanjutnya, berdasarkan Persamaan (3.31) diperoleh operasi pada lapisan *output:*

$$y = \sum_{k=1}^{4} v_{k1} T_k + b_0$$

$$= v_{11}T_1 + v_{21}T_2 + v_{31}T_3 + v_{41}T_4 + v_{51}T_5 + v_{61}T_6 + v_{71}T_7 + v_{81}T_8 + v_{91}T_9 + v_{10\_1}T_{10} + v_{11\_1}T_{11} + b_0$$

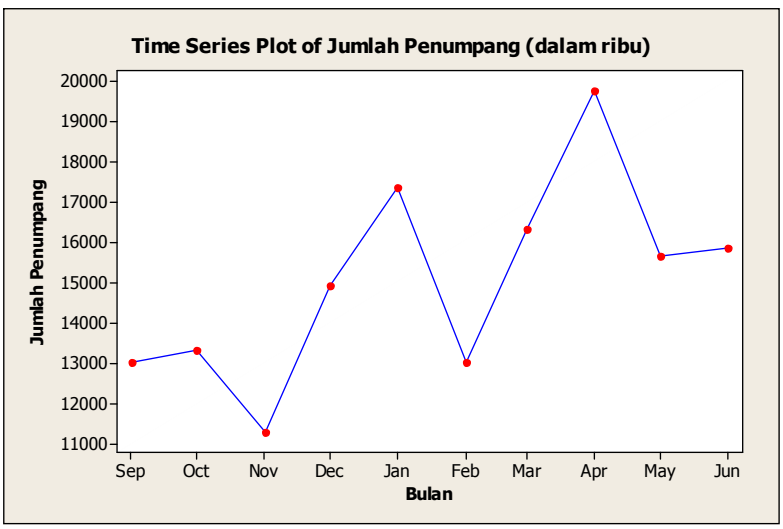
$$= (0{,}914)(-2{,}483) + (-0{,}891)(2{,}666) + (-0{,}109)(1{,}688) + (0{,}460)(0{,}942) + (-0{,}792)(-1{,}660) + (-0{,}845)(-3{,}005) + (-0{,}539)(-2{,}327) + (0{,}909)(2{,}019) + (0{,}986)(-2{,}433) + (-0{,}951)(-2{,}094) + (-0{,}6015)(-1{,}145) - 0{,}025 = 2{,}80243$$

Diperoleh nilai *y**=2,80243. Nilai tersebut masih dalam bentuk normal. Untuk mengembalikan dalam bentuk data semula, digunakan fungsi *poststd* pada *matlab.* Hasil peramalan penumpang kereta api Jabodetabek bulan September 2013 adalah 13011000. Untuk meramalkan jumlah penumpang bulan Oktober 2013 adalah dengan cara seperti semula yaitu dengan menambahkan peramalan jumlah penumpang terakhir (September 2013) sebagai *input*nya. Hasil peramalan 10 bulan berikutnya dalam bentuk data normalisasi terlihat pada tabel 3.15 berikut:

Tabel 3.15. Nilai Normal Peramalan

Waktu	Nilai Normal	Rata-Rata	Standar Deviasi	Nilai Denormalisasi (ribu)
September 2013	2.802	10304.848	965.584	13011
Oktober 2013	2.979	10333.946	1000.481	13314
November 2013	0.896	10365.649	1041.478	11299
Desember 2013	4.392	10375.474	1040.340	14944
Januari 2014	6.103	10423.062	1135.044	17351
Februari 2014	1.903	10494.484	1330.305	13026
Maret 2014	4.314	10520.316	1347.910	16335
April 2014	6.277	10579.050	1462.820	19761
Mei 2014	2.896	10670.870	1720.846	15654
Juni 2014	2.892	10720.208	1782.570	15876

Setelah dilakukan proses denormalisasi dengan bantuan program *Matlab R2009a*, diperoleh hasil peramalan penumpang yang tertulis pada Tabel 3.15. Pada tabel 3.15, hasil peramalan dengan model *recurrent neural network* 10 bulan pada Oktober 2013 sampai Juni 2014. Gambar 3.15 adalah plot data peramalan Oktober 2013 sampai Juni 2014.



Gambar 3.15. Plot Data Peramalan

Dari hasil peramalan, diperkirakan jumlah penumpang kereta api akan mencapai jumlah yang tinggi pada bulan Januari 2014 sebanyak 17351000 penumpang, Maret 2014 mencapai 16335000 penumpang dan yang tertinggi pada bulan April 2014 yaitu sebanyak 19761000 penumpang.

BAB IV
KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil pembahasan peramalan penumpang kereta api Jabodetabek menggunakan model *recurrent neural network* dan *recurrent neuro fuzzy* dapat disimpulkan hal-hal sebagai berikut:

1. Prosedur pemodelan *recurrent neural network* adalah sebagai berikut:
   a. Memilih *input* jaringan.

      Pemilihan *input* dan *output* jaringan dilakukan dengan melihat plot fungsi autokorelasi data. Variabel yang layak dijadikan *input* jaringan adalah *lag-lag* yang signifikan dari fungsi autokorelasi tersebut.

   b. Membagi data

      Data dibagi menjadi 2 yaitu data *training* dan data *testing*. Data *training* digunakan untuk pelatihan yang menghasilkan bobot-bobot jaringan sedangkan data *training* digunakan untuk pengujian atau validasi data. Ukuran pembagian data adalah 75% untuk data *training* dan 25% untuk data *testing*

   c. Normalisasi data

      Data dibawa dalam bentuk normal. Normalisasi data menggunakan *mean* dan standart *deviasi* yaitu dengan cara mengubah data asli menjadi data yang memiliki mean=0 dan standart *deviasi*=1

   d. Membentuk dan memilih struktur model terbaik

      Perancangan model ini dilakukan agar menghasilkan model peramalan terbaik dengan menentukan banyaknya neuron tersembunyi jaringan. Pencarian banyaknya neuron tersembunyi dilakukan dengan cara *trial* dan *error* terhadap beberapa jaringan yang mungkin. Sedangkan Pemilihan struktur dilakukan dengan mengeliminasi masing-masing *input* dan memilih jaringan dengan MAPE dan MSE terkecil. Perancangan serta pemilihan model terbaik menggunakan algoritma *bacpropagation*.

e. Denormalisasi data

   Data pembelajaran adalah data yang memiliki mean=0 dan standart deviasi=1. Setelah pembelajaran, data dikembalikan menjadi data asli.

f. Uji kesesuaian model

   Uji kesesuaian dilakukan dengan melihat plot fungsi autokorelasi dari *error* yang dihasilkan. Model yang baik adalah model dengan *error* yang *white noise*.

2. Presodur pemodelan *recurrent neuro fuzzy* adalah sebagai berikut:

   a. Memilih *input*

      Pemilihan *input* jaringan dilakukan dengan melihat plot fungsi autokorelasi data. Variabel yang layak dijadikan *input* jaringan adalah *lag-lag* yang signifikan dari fungsi autokorelasi tersebut.

   b. Membagi data

      Data dibagi menjadi 2 yaitu data *training* dan data *testing*. Ukuran pembagian data adalah 75% untuk data *training* dan 25% untuk data *testing*.

   c. Memilih struktur model terbaik dengan mencari jumlah neuron tersembunyi yang menghasilkan model terbaik. Setelah mendapatkan neuron tersembunyi, dilakukan eliminasi terhadap *input* jaringan. Pembelajaran menggunakan *recurren neural network* algoritma *backpropagation*.

   d. Membagi data ke dalam *cluster*.

      Jumlah *cluster* yang digunakan adalah 3. Metode yang digunakan adalah *Fuzzy C-Means* dimana proses *clustering* didasarkan nilai keanggotaan dari data.

   e. Melakukan pembelajaran jaringan syaraf yang berhubungan dengan anteseden (bagian *IF)* pada aturan-aturan inferensi *fuzzy* untuk mendapatkan nilai keanggotaan setiap data pada anteseden. Pembelajaran menggunakan jaringan *recurrent neural network* algoritma *backpropagation*.

f. Melakukan pembelajaran jaringan syaraf yang berhubungan dengan konsekuen (bagian *THEN*) pada aturan-aturan inferensi *fuzzy* yang digunakan untuk menentukan parameter pada setiap *cluster* dengan metode *Least Square Error* (LSE).

g. Menyederhanakan bagian konsekuen dengan melakukan eliminasi pada variabel menggunakan jaringan *recurrent neural network* algoritma *backpropagation* yang bertujuan untuk memperbaiki bagian anteseden. Sistem inferensi yang digunakan adalah sistem inferensi Soegeno orde 1, sehingga bagian konsekuen berbentuk persamaan linear.

h. Uji kesesuaian model

   Uji kesesuaian dilakukan dengan melihat plot fungsi autokorelasi dari *error* yang dihasilkan. Model yang baik adalah model dengan *error* yang *white noise*.

3. Dihasilkan 2 model peramalan yaitu model I model *recurrent neural network* dengan *input t-1, t-2, t-3,* dan *t-4* dan Model II model *recurrent neuro fuzzy* dengan *input t-1,t-2, t-3,* dan *t-4*. Terpilih model terbaik yang digunakan untuk meramalkan jumlah penumpang kereta api Jabodetabek yaitu model *recurrent neural network* dengan *input t-1, t-2, t-3,* dan *t-4*. Model ini menghasilkan MAPE *training* 1,2617% dan MSE *training* 29500000. Serta MAPE *testing* 3,785% dan MSE *testing* 2610000000. Model digunakan untuk peramalan penumpang kereta api Jabodetabek sepuluh bulan berikutnya.

B. SARAN

Dalam penulisan tugas akhir ini menggunakan *recurren neural network* dan *recurrent neuro fuzzy* untuk meramalkan penumpang kereta api Jabodetabek. Untuk penulisan tugas akhir yang lain, disarankan agar memilih model jaringan lain pada *recurrent neural network*. Jika pada tugas akhir ini menggunakan jaringan Elman, pada tugas akhir yang lain dapat menggunakan jaringan Hopfield. Kemudian untuk model *neuro fuzzy*, bisa menggunakan sistem inferensi yang berbeda. Misalkan sistem inferensi Tsukamoto. Dari

segi data, penulisan tugas akhir lainnya supaya lebih teliti memperhatikan pola-pola data serta juga memperhitungkan variabel-variabel lain.

DAFTAR PUSTAKA

Abbas Salim *Manajemen Transportasi* (2004). Jakarta: PT Raja Grafindo Persada.

Ali Zilouchian (2001). *Fundamental of Neural Netwoek*, by CRC Press LLC.

Bails, Dale G. (1993). *Business Fluctuations: Forecasting Techniques and Aplications.* Englewood Cliffs, New Jersey: Prentice-Hall.

Beale, Mark Hudson et al (2010). *Neural Network Toolbox TM 7 User's Guide.* Natick: The Math Works, Inc.

Dinas Perhubungan DKI Jakarta. *Pengembangan Sistem Transportasi Jakarta yang Terintegrasi dan Berkualitas untuk Mewujudkan Efisiensi Energi*. diakses di http://iesr.or.id/files/Pemprov%20DKI%20Jakarta.pdf 31 Februari 2014 jam 07.00.

Fausett, L. (1994). *Fundamentals of Neural Network (Archetectures, Algoruthms, and Applications).*Upper Saddle River, New-Jersey: Prentice-Hall.

Frechtling, Douglas C. (2001). *Forecasting Tourism Demand: Methods and Strategies.* London: Butterworth-Heinemann.

Gurney, Kevin. (1997). *An Introduction to Neural Networks*. London:UCL Press.

Hanke, J.E. & Wichern, D.W. (2005). *Bussiness Forecasting, 8th Edition*. Upper Saddle River, New Jersey:Prentice Hall.

http://id.wikipedia.org/wiki/Daerah_Operasi_I_Jakarta diakses tanggal 31 Januari 2014 08:38

http://id.wikipedia.org/wiki/Jabotabek diakses tanggal 9 Januari 2014 pukul 08:41.

Hu,Yu Hen dan Jenq-Neng Hwang. (2002). *Handbook of Neural Network Signal Processing.* Florida: CRC Press LLC.

Ibrahim, Ahmad M. (2004). *Fuzzy Logic for Embedded Systems Applications.* Burlington: Elsevier Science.

Klir G.J., St. Clair U., Yuan, B. (1997). *Fuzzy Set Theory Foundation and Application.* Upper Saddle River , New Jersey: Prentice Hall Inc.

Lin Chao. C et al, (2008). *Artificial Neural Network Prediction of Clozapine Response with Combine Pharmacogenetic and Clinical Data.*

Lin, C.-T., & Lee, G. (1996). *Neuro Fuzzy Systems.*Upper Saddle River, New Jersey: Prentice-Hall.

Makridakis, S, et al. (1999). *Metode dan Aplikasi Peramalan Jilid I (Ir. Untung Sus Ardiyanto, M.Sc. dan Ir. Abdul Basith, M.Sc. Terjemahan)*. Edisi Kedua. Jakarta: Penerbit Erlangga.

Montgomery, C.D, Cheryl, L. J, & Murat, K. (2007). *Introduction to time series analysis & Forecasting.* New York: Willey

Nur Khayati Setiyawati (2013). *Aplikasi Model Elman Recurrent Neural Network dengan Algoritma Backpropagation untuk Peramalan Indeks Harga Konsumen di Yogyakarta.* Skripsi. UNY.

Prabowo Pudjo Widodo, & Handayanto, R. T. (2012). *Penerapan Soft Computing Dengan MATLAB.* Bandung: Rekayasa Sains.

Ross, Timothy J. (2010). *Fuzzy Logic with Enginerring Applications.* Singapore: Fabulous Printers Pte Ltd.

Siang, J. J. (2009). *Jaringan Syaraf Tiruan & Pemrogramannya Menggunakan Matlab.* Yogyakarta: Andi.

Singgih Santoso (2009). *Business Forecasting (Metode Peramalan Bisnis Masa Kini dengan MINITAB dan SPSS)*. Jakarta : PT.Alex Media Komputindo

Sri Kusumadewi & Sri Hartati (2010). *Neuro Fuzzy: Integrasi Sistem Fuzzy & Jaringan Syaraf Edisi 2.* Yogyakarta: Graha Ilmu.

Sri Kusumadewi (2004). *Membangun Jaringan Syaraf Tiruan Menggunakan Matlab dan Excel Link*. Yogyakarta: Graha Ilmu.

Taufik Hidayat (2004). *Perkereta Apian Indonesia Di Persimpangan Jalan.* Jakarta: Yayasan Lembaga Konsumen Indonesia.

Tiara Anggraeni (2013). *Prediksi Suhu Udara di Yogyakarta dengan Model Neuro-Fuzzy.* Skripsi. UNY.

web.cecs.pdx.edu/...479/.../2012-1161.%20Neuro-Fuzzy%20Systems.pp...diakses tanggal 21 Januari pukul 16:00

Wei, W.W.S. 2006. *Time Series Analysis, Univariate and Multivariate Methods*. Boston: Addison Wesley Publishing Company.

www.bps.go.id diakses tanggal 3 Oktober 2013

Zimmermann. (1991). *Fuzzy Sets Theory and its Applications* (2 ed.). Massachusetts: Kluwer Academic Publishers.

LAMPIRAN

Lampiran 3.1 Data Jumlah Penumpang Kereta Api

No	Waktu	Jumlah Penumpang(ribu)
1	Januari 2006	8681
2	Februari 2006	8144
3	Maret 2006	8920
4	April 2006	8462
5	Mei 2006	8899
6	Juni 2006	8606
7	Juli 2006	8787
8	Agustus 2006	8661
9	September 2006	8829
10	Oktober 2006	8767
11	November 2006	8895
12	Desember 2006	8774
13	Januari 2007	8723
14	Februari 2007	7566
15	Maret 2007	9009
16	April 2007	10206
17	Mei 2007	10608
18	Juni 2007	10310
19	Juli 2007	10761
20	Agustus 2007	10653
21	September 2007	10446
22	Oktober 2007	9887
23	November 2007	10031
24	Desember 2007	9895
25	Januari 2008	9868
26	Februari 2008	9673
27	Maret 2008	10469
28	April 2008	10562
29	Mei 2008	10582
30	Juni 2008	10824
31	Juli 2008	11206
32	Agustus 2008	11129
33	September 2008	10481
34	Oktober 2008	10379
35	November 2008	10583
36	Desember 2008	9695
37	Januari 2009	10686
38	Februari 2009	9984

39	Maret 2009	11185
40	April 2009	10908
41	Mei 2009	11448
42	Juni 2009	11384
43	Juli 2009	11348
44	Agustus 2009	10905
45	September 2009	10243
46	Oktober 2009	11087
47	Oktober 2009	10592
48	Desember 2009	10738
49	Januari 2010	10541
50	Februari 2010	9641
51	Maret 2010	10759
52	April 2010	10394
53	Mei 2010	10476
54	Juni 2010	10312
55	Juli 2010	10466
56	Agustus 2010	10438
57	September 2010	9685
58	Oktober 2010	10796
59	November 2010	10106
60	Desember 2010	10694
61	Januari 2011	10354
62	Februari 2011	9270
63	Maret 2011	10733
64	April 2011	10188
65	Mei 2011	10513
66	Juni 2011	10147
67	Juli 2011	10749
68	Agustus 2011	9678
69	September 2011	9692
70	Oktober 2011	10152
71	November 2011	9852
72	Desember 2011	9777
73	Januari 2012	9779
74	Februari 2012	9840
75	Maret 2012	11285
76	April 2012	11271
77	Mei 2012	11872
78	Juni 2012	12034
79	Juli 2012	12391

80	Agustus 2012	11471
81	September 2012	11556
82	Oktober 2012	11501
83	November 2012	10650
84	Desember 2012	10438
85	Januari 2013	10089
86	Februari 2013	10281
87	Maret 2013	11240
88	April 2013	11529
89	Mei 2013	11767
90	Juni 2013	11517
91	Juli 2013	11906
92	Agustus 2013	11737

Lampiran 3.2 Data *Input* dan Data *Output*

No	***Input* (ribu)**				***Output* (ribu)**
	y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
1	8681	8144	8920	8462	8899
2	8144	8920	8462	8899	8606
3	8920	8462	8899	8606	8787
4	8462	8899	8606	8787	8661
5	8899	8606	8787	8661	8829
6	8606	8787	8661	8829	8767
7	8787	8661	8829	8767	8895
8	8661	8829	8767	8895	8774
9	8829	8767	8895	8774	8723
10	8767	8895	8774	8723	7566
11	8895	8774	8723	7566	9009
12	8774	8723	7566	9009	10206
13	8723	7566	9009	10206	10608
14	7566	9009	10206	10608	10310
15	9009	10206	10608	10310	10761
16	10206	10608	10310	10761	10653
17	10608	10310	10761	10653	10446
18	10310	10761	10653	10446	9887
19	10761	10653	10446	9887	10031
20	10653	10446	9887	10031	9895
21	10446	9887	10031	9895	9868

22	9887	10031	9895	9868	9673
23	10031	9895	9868	9673	10469
24	9895	9868	9673	10469	10562
25	9868	9673	10469	10562	10582
26	9673	10469	10562	10582	10824
27	10469	10562	10582	10824	11206
28	10562	10582	10824	11206	11129
29	10582	10824	11206	11129	10481
30	10824	11206	11129	10481	10379
31	11206	11129	10481	10379	10583
32	11129	10481	10379	10583	9695
33	10481	10379	10583	9695	10686
34	10379	10583	9695	10686	9984
35	10583	9695	10686	9984	11185
36	9695	10686	9984	11185	10908
37	10686	9984	11185	10908	11448
38	9984	11185	10908	11448	11384
39	11185	10908	11448	11384	11348
40	10908	11448	11384	11348	10905
41	11448	11384	11348	10905	10243
42	11384	11348	10905	10243	11087
43	11348	10905	10243	11087	10592
44	10905	10243	11087	10592	10738
45	10243	11087	10592	10738	10541
46	11087	10592	10738	10541	9641
47	10592	10738	10541	9641	10759
48	10738	10541	9641	10759	10394
49	10541	9641	10759	10394	10476
50	9641	10759	10394	10476	10312
51	10759	10394	10476	10312	10466
52	10394	10476	10312	10466	10438
53	10476	10312	10466	10438	9685
54	10312	10466	10438	9685	10796
55	10466	10438	9685	10796	10106
56	10438	9685	10796	10106	10694
57	9685	10796	10106	10694	10354
58	10796	10106	10694	10354	9270
59	10106	10694	10354	9270	10733
60	10694	10354	9270	10733	10188

61	10354	9270	10733	10188	10513
62	9270	10733	10188	10513	10147
63	10733	10188	10513	10147	10749
64	10188	10513	10147	10749	9678
65	10513	10147	10749	9678	9692
66	10147	10749	9678	9692	10152
67	10749	9678	9692	10152	9852
68	9678	9692	10152	9852	9777
69	9692	10152	9852	9777	9779
70	10152	9852	9777	9779	9840
71	9852	9777	9779	9840	11285
72	9777	9779	9840	11285	11271
73	9779	9840	11285	11271	11872
74	9840	11285	11271	11872	12034
75	11285	11271	11872	12034	12391
76	11271	11872	12034	12391	11471
77	11872	12034	12391	11471	11556
78	12034	12391	11471	11556	11501
79	12391	11471	11556	11501	10650
80	11471	11556	11501	10650	10438
81	11556	11501	10650	10438	10089
82	11501	10650	10438	10089	10281
83	10650	10438	10089	10281	11240
84	10438	10089	10281	11240	11529
85	10089	10281	11240	11529	11767
86	10281	11240	11529	11767	11517
87	11240	11529	11767	11517	11906
88	11529	11767	11517	11906	11737

Lampiran 3.3 Pembagian Data *Training* dan Data *Testing*

Pembagian Data	***Input* (ribu)**				***Output* (ribu)**
	y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
	8681	8144	8920	8462	8899
	8144	8920	8462	8899	8606
	8920	8462	8899	8606	8787
	8462	8899	8606	8787	8661
	8899	8606	8787	8661	8829
Training	8606	8787	8661	8829	8767

	8787	8661	8829	8767	8895
	8661	8829	8767	8895	8774
	8829	8767	8895	8774	8723
	8767	8895	8774	8723	7566
	8895	8774	8723	7566	9009
	8774	8723	7566	9009	10206
	8723	7566	9009	10206	10608
	7566	9009	10206	10608	10310
	9009	10206	10608	10310	10761
	10206	10608	10310	10761	10653
	10608	10310	10761	10653	10446
	10310	10761	10653	10446	9887
	10761	10653	10446	9887	10031
	10653	10446	9887	10031	9895
	10446	9887	10031	9895	9868
	9887	10031	9895	9868	9673
	10031	9895	9868	9673	10469
	9895	9868	9673	10469	10562
	9868	9673	10469	10562	10582
	9673	10469	10562	10582	10824
	10469	10562	10582	10824	11206
	10562	10582	10824	11206	11129
	10582	10824	11206	11129	10481
	10824	11206	11129	10481	10379
	11206	11129	10481	10379	10583
	11129	10481	10379	10583	9695
	10481	10379	10583	9695	10686
	10379	10583	9695	10686	9984
	10583	9695	10686	9984	11185
	9695	10686	9984	11185	10908
	10686	9984	11185	10908	11448
	9984	11185	10908	11448	11384
	11185	10908	11448	11384	11348
	10908	11448	11384	11348	10905
	11448	11384	11348	10905	10243
	11384	11348	10905	10243	11087
	11348	10905	10243	11087	10592
	10905	10243	11087	10592	10738
	10243	11087	10592	10738	10541

	11087	10592	10738	10541	9641
	10592	10738	10541	9641	10759
	10738	10541	9641	10759	10394
	10541	9641	10759	10394	10476
	9641	10759	10394	10476	10312
	10759	10394	10476	10312	10466
	10394	10476	10312	10466	10438
	10476	10312	10466	10438	9685
	10312	10466	10438	9685	10796
	10466	10438	9685	10796	10106
	10438	9685	10796	10106	10694
	9685	10796	10106	10694	10354
	10796	10106	10694	10354	9270
	10106	10694	10354	9270	10733
	10694	10354	9270	10733	10188
	10354	9270	10733	10188	10513
	9270	10733	10188	10513	10147
	10733	10188	10513	10147	10749
	10188	10513	10147	10749	9678
	10513	10147	10749	9678	9692
	10147	10749	9678	9692	10152
Testing	10749	9678	9692	10152	9852
	9678	9692	10152	9852	9777
	9692	10152	9852	9777	9779
	10152	9852	9777	9779	9840
	9852	9777	9779	9840	11285
	9777	9779	9840	11285	11271
	9779	9840	11285	11271	11872
	9840	11285	11271	11872	12034
	11285	11271	11872	12034	12391
	11271	11872	12034	12391	11471
	11872	12034	12391	11471	11556
	12034	12391	11471	11556	11501
	12391	11471	11556	11501	10650
	11471	11556	11501	10650	10438
	11556	11501	10650	10438	10089
	11501	10650	10438	10089	10281
	10650	10438	10089	10281	11240
	10438	10089	10281	11240	11529

	10089	10281	11240	11529	11767
	10281	11240	11529	11767	11517
	11240	11529	11767	11517	11906
	11529	11767	11517	11906	11737

Lampiran 3.4 Data *Target*, *Output*, dan *Error* Model RNN dalam ribuan

Waktu	**Target(data aktual)**	***Output* akhir**	***Error***
Mei 2006	8899	8863.96	35.0401
Juni 2006	8606	8648.24	-42.244
Juli 2006	8787	8862.39	-75.386
Agustus 2006	8661	8386.12	274.88
September 2006	8829	8882.35	-53.354
Oktober 2006	8767	8866.63	-99.628
November 2006	8895	8875.46	19.5422
Desember 2006	8774	8834.67	-60.673
Januari 2007	8723	8398.92	324.078
Februari 2007	7566	7941.84	-375.84
Maret 2007	9009	9017.23	-8.2342
April 2007	10206	10214.5	-8.4567
Mei 2007	10608	10610.1	-2.113
Juni 2007	10310	10302.1	7.87134
Juli 2007	10761	10777.9	-16.882
Agustus 2007	10653	10527.5	125.479
September 2007	10446	10412.3	33.7486
Oktober 2007	9887	9803.11	83.8876
November 2007	10031	10079.7	-48.703
Desember 2007	9895	9915.86	-20.861
Januari 2008	9868	10031.2	-163.16
Februari 2008	9673	10089.1	-416.11
Maret 2008	10469	10149.6	319.442
April 2008	10562	10467.6	94.3573
Mei 2008	10582	10589.2	-7.1717
Juni 2008	10824	10789	35.031
Juli 2008	11206	11016.7	189.252
Agustus 2008	11129	11173.7	-44.746
September 2008	10481	10624.3	-143.28

Oktober 2008	10379	10503	-123.97
November 2008	10583	10784.6	-201.57
Desember 2008	9695	9675.14	19.8612
Januari 2009	10686	10525.9	160.08
Februari 2009	9984	9746.71	237.285
Maret 2009	11185	10790.8	394.222
April 2009	10908	11017.8	-109.78
Mei 2009	11448	11458.9	-10.93
Juni 2009	11384	11382.9	1.05302
Juli 2009	11348	11210	138.045
Agustus 2009	10905	10833.8	71.238
September 2009	10243	10117.4	125.647
Oktober 2009	11087	10930.5	156.545
Oktober 2009	10592	10669.8	-77.836
Desember 2009	10738	10769.2	-31.231
Januari 2010	10541	10800.3	-259.25
Februari 2010	9641	9856.53	-215.53
Maret 2010	10759	10787.2	-28.153
April 2010	10394	10292	102.01
Mei 2010	10476	10354.2	121.832
Juni 2010	10312	10217	94.9837
Juli 2010	10466	10240.2	225.806
Agustus 2010	10438	10291.3	146.692
September 2010	9685	10061.8	-376.83
Oktober 2010	10796	10837.2	-41.183
November 2010	10106	10275	-168.97
Desember 2010	10694	10927.8	-233.79
Januari 2011	10354	10229.6	124.352
Februari 2011	9270	9666.42	-396.42
Maret 2011	10733	10734.6	-1.6153
April 2011	10188	10253.7	-65.688
Mei 2011	10513	10514.9	-1.8535
Juni 2011	10147	10113.9	33.1394
Juli 2011	10749	10328.1	420.912
Agustus 2011	9678	9637.85	40.1516
September 2011	9692	9830.6	-138.6
Oktober 2011	10152	10102.2	49.8262

November 2011	9852	10191.3	-339.3
Desember 2011	9777	10106.6	-329.55
Januari 2012	9779	10080.6	-301.6
Februari 2012	9840	9482.64	357.36
Maret 2012	11285	9498.59	1786.41
April 2012	11271	11346.1	-75.068
Mei 2012	11872	9903.55	1968.45
Juni 2012	12034	9245.33	2788.67
Juli 2012	12391	10162.4	2228.56
Agustus 2012	11471	8895.27	2575.73
September 2012	11556	12383.3	-827.32
Oktober 2012	11501	13914.7	-2413.7
November 2012	10650	11811.7	-1161.7
Desember 2012	10438	10871.1	-433.14
Januari 2013	10089	11568.6	-1479.6
Februari 2013	10281	10578	-296.98
Maret 2013	11240	10058.5	1181.47
April 2013	11529	11295.8	233.178
Mei 2013	11767	9200.96	2566.04
Juni 2013	11517	8783.14	2733.86
Juli 2013	11906	9776.7	2129.3
Agustus 2013	11737	11354.1	382.897

Lampiran 3.5 Pembagian *Cluster*

Waktu	**Data (ribu)**	***Cluster***
Mei 2006	8899	3
Juni 2006	8606	3
Juli 2006	8787	3
Agustus 2006	8661	3
September 2006	8829	3
Oktober 2006	8767	3
November 2006	8895	3
Desember 2006	8774	3
Januari 2007	8723	3
Februari 2007	7566	3
Maret 2007	9009	3
April 2007	10206	3

Mei 2007	10608	3
Juni 2007	10310	3
Juli 2007	10761	2
Agustus 2007	10653	1
September 2007	10446	1
Oktober 2007	9887	2
November 2007	10031	2
Desember 2007	9895	2
Januari 2008	9868	2
Februari 2008	9673	2
Maret 2008	10469	2
April 2008	10562	2
Mei 2008	10582	2
Juni 2008	10824	2
Juli 2008	11206	1
Agustus 2008	11129	1
September 2008	10481	1
Oktober 2008	10379	1
November 2008	10583	1
Desember 2008	9695	2
Januari 2009	10686	2
Februari 2009	9984	2
Maret 2009	11185	1
April 2009	10908	1
Mei 2009	11448	1
Juni 2009	11384	1
Juli 2009	11348	1
Agustus 2009	10905	1
September 2009	10243	1
Oktober 2009	11087	1
Oktober 2009	10592	1
Desember 2009	10738	1
Januari 2010	10541	1
Februari 2010	9641	1
Maret 2010	10759	2
April 2010	10394	2
Mei 2010	10476	2
Juni 2010	10312	2

Juli 2010	10466	1
Agustus 2010	10438	2
September 2010	9685	2
Oktober 2010	10796	2
November 2010	10106	2
Desember 2010	10694	2
Januari 2011	10354	2
Februari 2011	9270	2
Maret 2011	10733	2
April 2011	10188	2
Mei 2011	10513	2
Juni 2011	10147	2
Juli 2011	10749	1
Agustus 2011	9678	2
September 2011	9692	2
Oktober 2011	10152	2
November 2011	9852	2
Desember 2011	9777	2
Januari 2012	9779	2
Februari 2012	9840	2
Maret 2012	11285	2
April 2012	11271	1
Mei 2012	11872	1
Juni 2012	12034	1
Juli 2012	12391	3
Agustus 2012	11471	3
September 2012	11556	3
Oktober 2012	11501	3
November 2012	10650	3
Desember 2012	10438	3
Januari 2013	10089	3
Februari 2013	10281	2
Maret 2013	11240	2
April 2013	11529	1
Mei 2013	11767	1
Juni 2013	11517	1
Juli 2013	11906	3
Agustus 2013	11737	3

Lampiran 3.6 Nilai Keanggotaan

Waktu	Data(ribu)	Nilai Keanggotaan		
Mei 2006	8899	0.0115	-0.0354	1.00052
Juni 2006	8606	0.01577	-0.0204	1.00462
Juli 2006	8787	0.00029	-0.0026	0.99618
Agustus 2006	8661	0.0076	-0.0056	1.0031
September 2006	8829	0.00019	0.00314	0.99666
Oktober 2006	8767	0.00439	-0.0022	1.00273
November 2006	8895	-0.0004	0.00311	1.00005
Desember 2006	8774	-0.0012	0.00807	1.00373
Januari 2007	8723	-0.0058	0.01605	0.9988
Februari 2007	7566	-0.003	0.01497	0.99995
Maret 2007	9009	0.00445	-0.0084	0.98979
April 2007	10206	0.0117	-0.0057	0.99075
Mei 2007	10608	-0.0099	-0.0097	1.00099
Juni 2007	10310	-0.0013	0.01533	1.00764
Juli 2007	10761	-0.0657	1.00679	0.04617
Agustus 2007	10653	0.51667	0.48734	-0.0117
September 2007	10446	1.0449	-0.0304	-0.0171
Oktober 2007	9887	0.12942	0.87267	0.0075
November 2007	10031	0.08723	0.90911	0.00134
Desember 2007	9895	0.07001	0.94429	-0.0181
Januari 2008	9868	0.15114	0.86579	-0.0405
Februari 2008	9673	-0.0797	1.04797	0.02264
Maret 2008	10469	-0.063	1.02057	0.02471
April 2008	10562	-0.1384	1.11281	0.02808
Mei 2008	10582	-0.0535	1.04631	-0.0243
Juni 2008	10824	0.16942	0.8414	-0.0138
Juli 2008	11206	1.10729	-0.0987	-0.0197
Agustus 2008	11129	1.01989	-0.0075	0.0028
September 2008	10481	0.90894	0.10854	0.01657
Oktober 2008	10379	1.0025	0.00743	0.0111
November 2008	10583	1.10845	-0.1343	0.02913
Desember 2008	9695	0.13266	0.8597	0.00458
Januari 2009	10686	-0.0474	1.05265	-0.0012
Februari 2009	9984	-0.1073	1.11241	0.01055
Maret 2009	11185	0.73912	0.2628	-0.0135
April 2009	10908	1.19204	-0.1845	-0.0123

Mei 2009	11448	1.02241	-0.0183	0.02808
Juni 2009	11384	0.92918	0.06455	0.0092
Juli 2009	11348	1.05833	-0.0774	0.03025
Agustus 2009	10905	0.9287	0.04363	-0.0374
September 2009	10243	0.98892	-0.0002	-0.0429
Oktober 2009	11087	1.04925	-0.0513	0.01295
Oktober 2009	10592	0.94998	0.01258	0.0306
Desember 2009	10738	1.0414	-0.0272	0.01777
Januari 2010	10541	1.05541	-0.0528	-0.0118
Februari 2010	9641	0.78182	0.22361	0.00539
Maret 2010	10759	-0.0361	1.04755	-0.0058
April 2010	10394	0.01946	0.98208	0.00966
Mei 2010	10476	-0.026	1.03927	-0.0282
Juni 2010	10312	0.00506	1.01587	-0.0182
Juli 2010	10466	0.6782	0.32004	0.00504
Agustus 2010	10438	0.01906	0.98477	-0.005
September 2010	9685	0.25686	0.7465	-0.0048
Oktober 2010	10796	0.01472	0.99892	-0.0174
November 2010	10106	0.08956	0.92045	0.00357
Desember 2010	10694	0.38876	0.60722	-0.0021
Januari 2011	10354	-0.0396	1.0613	-0.0107
Februari 2011	9270	0.20026	0.80962	-0.0031
Maret 2011	10733	0.06878	0.93921	-0.0046
April 2011	10188	-0.0216	1.03618	0.01067
Mei 2011	10513	-0.0452	1.04894	-0.0397
Juni 2011	10147	-0.0392	1.04252	0.00737
Juli 2011	10749	0.77899	0.2226	-0.003
Agustus 2011	9678	0.17403	0.83305	-0.0088
September 2011	9692	-0.1195	1.11363	0.02065
Oktober 2011	10152	0.03047	0.97129	0.01597
November 2011	9852	-0.0162	0.32016	0.71758
Desember 2011	9777	-0.0333	0.09116	0.96635
Januari 2012	9779	-0.0285	0.09851	0.98104
Februari 2012	9840	-0.0374	0.13981	0.93767
Maret 2012	11285	-0.0275	0.08602	0.9753
April 2012	11271	-0.0645	0.30135	0.81541
Mei 2012	11872	-0.0673	0.87726	0.12782
Juni 2012	12034	1.33143	-0.3303	-0.02
Juli 2012	12391	1.06022	-0.0446	0.01625

Agustus 2012	11471	0.91428	0.085	0.0056
September 2012	11556	1.10115	-0.1049	0.00451
Oktober 2012	11501	1.93077	-0.9419	-0.0399
November 2012	10650	0.9834	0.00071	0.02567
Desember 2012	10438	-0.1156	1.12057	0.01428
Januari 2013	10089	0.08213	0.9235	-0.0136
Februari 2013	10281	0.09629	0.90257	-0.0213
Maret 2013	11240	-0.0668	0.697	0.39824
April 2013	11529	-0.0368	0.85312	0.17439
Mei 2013	11767	0.16261	0.78281	0.01272
Juni 2013	11517	1.41295	-0.4075	-0.0228
Juli 2013	11906	1.28591	-0.2818	0.01152
Agustus 2013	11737	1.25685	-0.2481	-0.0088

Lampiran 3.7 Nilai Keanggotaan Himpunan Klasik Data *Training*

Waktu	**Data (ribu)**	**Nilai Keanggotaan (*Fuzzy*)**			**Nilai Keanggotaan Himpunan Klasik**		
		***Cluster* 1**	***Cluster* 2**	***Cluster* 3**	***Cluster* 1**	***Cluster* 2**	***Cluster* 3**
Mei 2006	8899	0.0115	-0.0354	1.00052	0	0	1
Juni 2006	8606	0.01577	-0.0204	1.00462	0	0	1
Juli 2006	8787	0.00029	-0.0026	0.99618	0	0	1
Agustus 2006	8661	0.0076	-0.0056	1.0031	0	0	1
September 2006	8829	0.00019	0.0031	0.99666	0	0	1
Oktober 2006	8767	0.00439	-0.0022	1.00273	0	0	1
November 2006	8895	-0.0004	0.00311	1.00005	0	0	1
Desember 2006	8774	-0.0012	0.00807	1.00373	0	0	1
Januari 2007	8723	-0.0058	0.01605	0.9988	0	0	1
Februari 2007	7566	-0.003	0.01497	0.99995	0	0	1
Maret 2007	9009	0.00445	-0.0084	0.98979	0	0	1
April 2007	10206	0.0117	-0.0057	0.99075	0	0	1
Mei 2007	10608	-0.0099	-0.0097	1.00099	0	0	1
Juni 2007	10310	-0.0013	0.01533	1.00764	0	0	1
Juli 2007	10761	-0.0657	1.00679	0.04617	0	1	0
Agustus 2007	10653	0.51667	0.48734	-0.0117	1	0	0
September 2007	10446	1.0449	-0.0304	-0.0171	1	0	0
Oktober 2007	9887	0.12942	0.87267	0.0075	0	1	0
November 2007	10031	0.08723	0.90911	0.00134	0	1	0

Desember 2007	9895	0.07001	0.94429	-0.0181	0	1	0
Januari 2008	9868	0.15114	0.86579	-0.0405	0	1	0
Februari 2008	9673	-0.0797	1.04797	0.02264	0	1	0
Maret 2008	10469	-0.063	1.02057	0.02471	0	1	0
April 2008	10562	-0.1384	1.11281	0.02808	0	1	0
Mei 2008	10582	-0.0535	1.04631	-0.0243	0	1	0
Juni 2008	10824	0.16942	0.8414	-0.0138	0	1	0
Juli 2008	11206	1.10729	-0.0987	-0.0197	1	0	0
Agustus 2008	11129	1.01989	-0.0075	0.0028	1	0	0
September 2008	10481	0.90894	0.10854	0.01657	1	0	0
Oktober 2008	10379	1.0025	0.00743	0.0111	1	0	0
November 2008	10583	1.10845	-0.1343	0.02913	1	0	0
Desember 2008	9695	0.13266	0.8597	0.00458	0	1	0
Januari 2009	10686	-0.0474	1.05265	-0.0012	0	1	0
Februari 2009	9984	-0.1073	1.11241	0.01055	0	1	0
Maret 2009	11185	0.73912	0.2628	-0.0135	1	0	0
April 2009	10908	1.19204	-0.1845	-0.0123	1	0	0
Mei 2009	11448	1.02241	-0.0183	0.02808	1	0	0
Juni 2009	11384	0.92918	0.06455	0.0092	1	0	0
Juli 2009	11348	1.05833	-0.0774	0.03025	1	0	0
Agustus 2009	10905	0.9287	0.04363	-0.0374	1	0	0
September 2009	10243	0.98892	-0.0002	-0.0429	1	0	0
Oktober 2009	11087	1.04925	-0.0513	0.01295	1	0	0
Oktober 2009	10592	0.94998	0.01258	0.0306	1	0	0
Desember 2009	10738	1.0414	-0.0272	0.01777	1	0	0
Januari 2010	10541	1.05541	-0.0528	-0.0118	1	0	0
Februari 2010	9641	0.78182	0.22361	0.00539	1	0	0
Maret 2010	10759	-0.0361	1.04755	-0.0058	0	1	0
April 2010	10394	0.01946	0.98208	0.00966	0	1	0
Mei 2010	10476	-0.026	1.03927	-0.0282	0	1	0
Juni 2010	10312	0.00506	1.01587	-0.0182	0	1	0
Juli 2010	10466	0.6782	0.32004	0.00504	1	0	0
Agustus 2010	10438	0.01906	0.98477	-0.005	0	1	0
September 2010	9685	0.25686	0.7465	-0.0048	0	1	0
Oktober 2010	10796	0.01472	0.99892	-0.0174	0	1	0
November 2010	10106	0.08956	0.92045	0.00357	0	1	0
Desember 2010	10694	0.38876	0.60722	-0.0021	0	1	0
Januari 2011	10354	-0.0396	1.0613	-0.0107	0	1	0
Februari 2011	9270	0.20026	0.80962	-0.0031	0	1	0

Maret 2011	10733	0.06878	0.93921	-0.0046	0	1	0
April 2011	10188	-0.0216	1.03618	0.01067	0	1	0
Mei 2011	10513	-0.0452	1.04894	-0.0397	0	1	0
Juni 2011	10147	-0.0392	1.04252	0.00737	0	1	0
Juli 2011	10749	0.77899	0.2226	-0.003	1	0	0
Agustus 2011	9678	0.17403	0.83305	-0.0088	0	1	0
September 2011	9692	-0.1195	1.11363	0.02065	0	1	0
Oktober 2011	10152	0.03047	0.97129	0.01597	0	1	0

Lampiran 3.8 Nilai Keanggotaan Himpunan Klasik Data *Testing*

Waktu	**Data (ribu)**	**Nilai Keanggotaan (*Fuzzy*)**			**Nilai Keanggotaan Himpunan Klasik**		
		***Cluster* 1**	***Cluster* 2**	***Cluster* 3**	***Cluster* 1**	***Cluster* 2**	***Cluster* 3**
November 2011	9852	-0.0162	0.32016	0.71758	0	1	0
Desember 2011	9777	-0.0333	0.09116	0.96635	0	1	0
Januari 2012	9779	-0.0285	0.09851	0.98104	0	1	0
Februari 2012	9840	-0.0374	0.13981	0.93767	0	1	0
Maret 2012	11285	-0.0275	0.08602	0.9753	0	1	0
April 2012	11271	-0.0645	0.30135	0.81541	1	0	0
Mei 2012	11872	-0.0673	0.87726	0.12782	1	0	0
Juni 2012	12034	1.33143	-0.3303	-0.02	1	0	0
Juli 2012	12391	1.06022	-0.0446	0.01625	0	0	1
Agustus 2012	11471	0.91428	0.085	0.0056	0	0	1
September 2012	11556	1.10115	-0.1049	0.00451	0	0	1
Oktober 2012	11501	1.93077	-0.9419	-0.0399	0	0	1
November 2012	10650	0.9834	0.00071	0.02567	0	0	1
Desember 2012	10438	-0.1156	1.12057	0.01428	0	0	1
Januari 2013	10089	0.08213	0.9235	-0.0136	0	0	1
Februari 2013	10281	0.09629	0.90257	-0.0213	0	1	0
Maret 2013	11240	-0.0668	0.697	0.39824	0	1	0
April 2013	11529	-0.0368	0.85312	0.17439	1	0	0
Mei 2013	11767	0.16261	0.78281	0.01272	1	0	0
Juni 2013	11517	1.41295	-0.4075	-0.0228	1	0	0
Juli 2013	11906	1.28591	-0.2818	0.01152	0	0	1
Agustus 2013	11737	1.25685	-0.2481	-0.0088	0	0	1

Lampiran 3.9 Data *Traing Cluster* 1

***Input* (ribu)**				**Target (ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
10206	10608	10310	10761	10653
10608	10310	10761	10653	10446
10469	10562	10582	10824	11206
10562	10582	10824	11206	11129
10582	10824	11206	11129	10481
10824	11206	11129	10481	10379
11206	11129	10481	10379	10583
10583	9695	10686	9984	11185
9695	10686	9984	11185	10908
10686	9984	11185	10908	11448
9984	11185	10908	11448	11384
11185	10908	11448	11384	11348
10908	11448	11384	11348	10905
11448	11384	11348	10905	10243
11384	11348	10905	10243	11087
11348	10905	10243	11087	10592
10905	10243	11087	10592	10738
10243	11087	10592	10738	10541
11087	10592	10738	10541	9641
10759	10394	10476	10312	10466
10733	10188	10513	10147	10749

Lampiran 3.10 Data *Testing Cluster* 1

Input				**target(t)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
9777	9779	9840	11285	11271
9779	9840	11285	11271	11872
9840	11285	11271	11872	12034
10438	10089	10281	11240	11529
10089	10281	11240	11529	11767

Lampiran 3.11 Data *Traing Cluster* 2

***Input* (ribu)**				***Target* (ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
9009	10206	10608	10310	10761
10310	10761	10653	10446	9887
10761	10653	10446	9887	10031
10653	10446	9887	10031	9895
10446	9887	10031	9895	9868
9887	10031	9895	9868	9673
10031	9895	9868	9673	10469
9895	9868	9673	10469	10562
9868	9673	10469	10562	10582
9673	10469	10562	10582	10824
11129	10481	10379	10583	9695
10481	10379	10583	9695	10686
10379	10583	9695	10686	9984
10592	10738	10541	9641	10759
10738	10541	9641	10759	10394
10541	9641	10759	10394	10476
9641	10759	10394	10476	10312
10394	10476	10312	10466	10438
10476	10312	10466	10438	9685
10312	10466	10438	9685	10796
10466	10438	9685	10796	10106
10438	9685	10796	10106	10694
9685	10796	10106	10694	10354
10796	10106	10694	10354	9270
10106	10694	10354	9270	10733
10694	10354	9270	10733	10188
10354	9270	10733	10188	10513
9270	10733	10188	10513	10147
10188	10513	10147	10749	9678
10513	10147	10749	9678	9692
10147	10749	9678	9692	10152

Lampiran 3.12 Data *Testing Cluster* 2

***Input* (ribu)**				**Target (ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
10749	9678	9692	10152	9852
9678	9692	10152	9852	9777
9692	10152	9852	9777	9779
10152	9852	9777	9779	9840
9852	9777	9779	9840	11285
11501	10650	10438	10089	10281

Lampiran 3.13 Data *Traing Cluster* 3

***Input* (ribu)**				**Target (ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
8681	8144	8920	8462	8899
8144	8920	8462	8899	8606
8920	8462	8899	8606	8787
8462	8899	8606	8787	8661
8899	8606	8787	8661	8829
8606	8787	8661	8829	8767
8787	8661	8829	8767	8895
8661	8829	8767	8895	8774
8829	8767	8895	8774	8723
8767	8895	8774	8723	7566
8895	8774	8723	7566	9009
8774	8723	7566	9009	10206
8723	7566	9009	10206	10608
7566	9009	10206	10608	10310

Lampiran 3.14 Data *Testing Cluster* 3

***Input* (ribu)**				**Target (ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
11285	11271	11872	12034	12391
11271	11872	12034	12391	11471
11872	12034	12391	11471	11556
12034	12391	11471	11556	11501
12391	11471	11556	11501	10650

11471	11556	11501	10650	10438
11556	11501	10650	10438	10089
11240	11529	11767	11517	11906
11529	11767	11517	11906	11737

Lampiran 3.15 Hasil Penyederhanaan *Training Cluster* 1

***Input* (ribu)**			**Hasil Penyederhanaan(ribu)**
y_{t-3}	y_{t-2}	y_{t-1}	
10608	10310	10761	10540.4859
10310	10761	10653	10736.3762
10562	10582	10824	10723.2202
10582	10824	11206	11097.5234
10824	11206	11129	11189.3563
11206	11129	10481	10645.6422
11129	10481	10379	10274.7498
9695	10686	9984	10348.616
10686	9984	11185	10657.9049
9984	11185	10908	11171.2297
11185	10908	11448	11200.7738
10908	11448	11384	11466.2672
11448	11384	11348	11318.9892
11384	11348	10905	11010.5697
11348	10905	10243	10350.674
10905	10243	11087	10679.644
10243	11087	10592	10864.4013
11087	10592	10738	10580.3364
10592	10738	10541	10601.0251
10394	10476	10312	10351.4792
10188	10513	10147	10291.9996

Lampiran 3.16 Hasil Penyederhanaan *Testing Cluster* 1

***Input* (ribu)**			**Hasil Penyederhanaan(ribu)**
y_{t-3}	y_{t-2}	y_{t-1}	
9779	9840	11285	10810.1603
9840	11285	11271	11491.5228
11285	11271	11872	11648.8289
10089	10281	11240	10940.8811

10281	11240	11529	11570.5779
11240	11529	11767	11710.1408

Lampiran 3.17 Hasil Penyederhanaan *Training Cluster* 2

***Input* (ribu)**			**Hasil Penyederhanaan(ribu)**
y_{t-4}	y_{t-3}	y_{t-1}	
9009	10206	10310	10007.6
10310	10761	10446	10538.8
10761	10653	9887	10369.9
10653	10446	10031	10296.6
10446	9887	9895	9937.33
9887	10031	9868	9911.93
10031	9895	9673	9801.46
9895	9868	10469	10031.4
9868	9673	10562	9960.04
9673	10469	10582	10336
11129	10481	10583	10573.8
10481	10379	9695	10123.7
10379	10583	10686	10540.2
10592	10738	9641	10303.9
10738	10541	10759	10600.4
10541	9641	10394	9994.93
9641	10759	10476	10441.5
10394	10476	10466	10415.5
10476	10312	10438	10336.7
10312	10466	9685	10137.3
10466	10438	10796	10517.7
10438	9685	10106	9904.75
9685	10796	10694	10539.7
10796	10106	10354	10256
10106	10694	9270	10081
10694	10354	10733	10490.7
10354	9270	10188	9710.02
9270	10733	10513	10381.8
10188	10513	10749	10495.6
10513	10147	9678	10006.4
10147	10749	9692	10255.7

Lampiran 3.18 Hasil Penyederhanaan *Testing Cluster* 2

***Input* (ribu)**			**Hasil Penyederhanaan(ribu)**
y_{t-4}	y_{t-3}	y_{t-1}	
10749	9678	10152	9965.97
9678	9692	9852	9702.9
9692	10152	9777	9911.48
10152	9852	9779	9834.37
9852	9777	9840	9769.3
11501	10650	10089	10553.3
10650	10438	10281	10375.4

Lampiran 3.19 Hasil Penyederhanaan *Training Cluster* 3

***Input* (ribu)**				**Hasil Penyederhanaan(ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
8681	8144	8920	8462	8736.03
8144	8920	8462	8899	8798.91
8920	8462	8899	8606	8915.44
8462	8899	8606	8787	8821.23
8899	8606	8787	8661	8943.84
8606	8787	8661	8829	8946.17
8787	8661	8829	8767	8972.45
8661	8829	8767	8895	9007.04
8829	8767	8895	8774	8962.5
8767	8895	8774	8723	8875.28
8895	8774	8723	7566	7863.51
8774	8723	7566	9009	9403.41
8723	7566	9009	10206	10548.8
7566	9009	10206	10608	9841.13

Lampiran 3.20 Hasil Penyederhanaan *Testing Cluster* 3

***Input* (ribu)**				**Hasil Penyederhanaan(ribu)**
y_{t-4}	y_{t-3}	y_{t-2}	y_{t-1}	
11285	11271	11872	12034	12137.9
11271	11872	12034	12391	12303.9
11872	12034	12391	11471	11604.6
12034	12391	11471	11556	11838.3

12391	11471	11556	11501	12155.7
11471	11556	11501	10650	10895.8
11556	11501	10650	10438	10893.5
11240	11529	11767	11517	11578.4
11529	11767	11517	11906	12075

Lampiran 3.21 Data *Target*, *Output*, dan *Error* Model RNF dalam ribu

No	Waktu	Target (Data aktual)	*Output* Peramalan	*Error*
1	Mei 2006	8899	7985.80041	913.1995904
2	Juni 2006	8606	8323.34817	282.6518302
3	Juli 2006	8787	8057.864676	729.1353243
4	Agustus 2006	8661	8235.624912	425.3750878
5	September 2006	8829	8117.661036	711.3389636
6	Oktober 2006	8767	8244.118983	522.8810172
7	November 2006	8895	8167.919125	727.0808753
8	Desember 2006	8774	8254.553962	519.4460384
9	Januari 2007	8723	8157.756161	565.2438385
10	Februari 2007	7566	8160.960252	-594.9602515
11	Maret 2007	9009	7577.95371	1431.04629
12	April 2007	10206	8748.340968	1457.659032
13	Mei 2007	10608	9117.354471	1490.645529
14	Juni 2007	10310	9007.167115	1302.832885
15	Juli 2007	10761	10163.51081	597.4891905
16	Agustus 2007	10653	10588.27819	64.72180611
17	September 2007	10446	10151.4824	294.5175992
18	Oktober 2007	9887	10454.62598	-567.6259821
19	November 2007	10031	10392.12766	-361.1276634
20	Desember 2007	9895	10378.95452	-483.9545236
21	Januari 2008	9868	10146.05694	-278.0569449
22	Februari 2008	9673	10147.5525	-474.5525039
23	Maret 2008	10469	10089.0028	379.997204
24	April 2008	10562	10012.40125	549.5987532
25	Mei 2008	10582	10207.57939	374.4206072
26	Juni 2008	10824	10342.64934	481.3506625
27	Juli 2008	11206	10408.05728	797.9427168
28	Agustus 2008	11129	10717.97014	411.0298587

29	September 2008	10481	10922.0483	-441.048296
30	Oktober 2008	10379	10317.69927	61.30072692
31	November 2008	10583	11131.37688	-548.3768807
32	Desember 2008	9695	10440.51164	-745.5116441
33	Januari 2009	10686	10258.71374	427.2862635
34	Februari 2009	9984	10389.28122	-405.281219
35	Maret 2009	11185	10142.1099	1042.890098
36	April 2009	10908	11608.0015	-700.0015031
37	Mei 2009	11448	10727.27436	720.7256415
38	Juni 2009	11384	11449.36667	-65.3666721
39	Juli 2009	11348	11253.63514	94.36485853
40	Agustus 2009	10905	11063.2418	-158.2417961
41	September 2009	10243	10448.31965	-205.319649
42	Oktober 2009	11087	10361.9342	725.0657975
43	Oktober 2009	10592	11596.38294	-1004.382941
44	Desember 2009	10738	10506.29187	231.7081314
45	Januari 2010	10541	11144.91145	-603.9114492
46	Februari 2010	9641	10402.26688	-761.2668803
47	Maret 2010	10759	10350.84538	408.1546171
48	April 2010	10394	10510.32933	-116.329329
49	Mei 2010	10476	10238.16788	237.8321168
50	Juni 2010	10312	10437.84567	-125.8456659
51	Juli 2010	10466	10139.73715	326.2628506
52	Agustus 2010	10438	10408.36425	29.63575145
53	September 2010	9685	10226.63953	-541.639525
54	Oktober 2010	10796	10297.56168	498.4383165
55	November 2010	10106	10537.74416	-431.7441635
56	Desember 2010	10694	10201.24409	492.7559121
57	Januari 2011	10354	10437.49247	-83.49246581
58	Februari 2011	9270	10259.69276	-989.6927587
59	Maret 2011	10733	10266.67351	466.3264912
60	April 2011	10188	10432.81927	-244.8192672
61	Mei 2011	10513	10084.93612	428.0638797
62	Juni 2011	10147	10345.12593	-198.1259307
63	Juli 2011	10749	9981.180777	767.8192226
64	Agustus 2011	9678	10518.18021	-840.1802053
65	September 2011	9692	10146.5969	-454.5968964

66	Oktober 2011	10152	10309.63456	-157.6345618
67	November 2011	9852	8954.163826	897.8361737
68	Desember 2011	9777	8224.749835	1552.250165
69	Januari 2012	9779	8280.495371	1498.504629
70	Februari 2012	9840	8333.427159	1506.572841
71	Maret 2012	11285	8293.386318	2991.613682
72	April 2012	11271	9158.989766	2112.010234
73	Mei 2012	11872	9768.508345	2103.491655
74	Juni 2012	12034	10455.69899	1578.301014
75	Juli 2012	12391	10982.1995	1408.800495
76	Agustus 2012	11471	11352.75744	118.2425636
77	September 2012	11556	10252.4293	1303.570698
78	Oktober 2012	11501	11964.58696	-463.5869575
79	November 2012	10650	10470.72105	179.2789503
80	Desember 2012	10438	10380.04104	57.9589644
81	Januari 2013	10089	10469.48705	-380.4870532
82	Februari 2013	10281	10123.65408	157.3459193
83	Maret 2013	11240	9449.944787	1790.055213
84	April 2013	11529	9842.608425	1686.391575
85	Mei 2013	11767	10250.96371	1516.036294
86	Juni 2013	11517	10481.35651	1035.643492
87	Juli 2013	11906	10416.08768	1489.912322
88	Agustus 2013	11737	11367.1647	369.8353016

Lampiran 3.22 Nilai $f_s(x_i)$ dalam ribu

Data ke-	*Cluster*			Data ke	*Cluster*		
	1	2	3		1	2	3
1	11700.6	9480.1	7996.01	45	11098.2	10579.9	9505.22
2	11298.6	9622.32	8303.01	46	10377.7	10505.6	9679.28
3	11762.7	9551.72	8060.62	47	10450.5	10344.4	8559.69
4	11303.7	9623.86	8220.08	48	11232.1	10502.1	9896.92
5	11694.8	9581.35	8112.38	49	10149.3	10215.8	9498.59
6	11516.4	9617.55	8232.87	50	10805	10412.5	9122.64
7	11675.8	9597.58	8164.76	51	10025.7	10393.5	9371.16
8	11494.1	9638	8247.33	52	10150.6	10408.4	9416.08
9	11553.7	9620.7	8153.84	53	9787.85	10372.4	9422.73
10	11307.3	9634.22	8148.47	54	10422.3	10266	8591.94

11	11073.2	9499.63	7578.48	55	11341.5	10462.2	9851.15
12	11879.9	9640.67	8716.51	56	10246	10168.9	9158.57
13	13516.5	9582.9	9165.54	57	11385.6	10461.9	9353.35
14	12067.5	9865.17	8997.96	58	9934.75	10337	9453.48
15	9917.58	10204.2	8925.58	59	10616.9	10230.9	8213.37
16	10674.7	10473.6	9622.87	60	11037.9	10451.1	9883.15
17	10151.4	10420	9672.95	61	10690.6	10081.7	9309.88
18	10461.5	10463.7	9285.12	62	11083	10381.9	9064.87
19	10532.6	10380.9	8872.21	63	9882.67	10315.4	9205.01
20	10465.7	10348.6	9126.27	64	10808.8	10448.2	9637.47
21	9631.51	10180.2	8955.64	65	10464	10208.7	8637.33
22	9957.73	10162.3	8795.16	66	10865.4	10318.3	8723.02
23	9883.9	10110.8	8667.74	67	11544.9	9826.74	8623.22
24	11613.3	10226.9	9398.67	68	11238	9695.02	8189.82
25	10326	10195.9	9445.37	69	10200.4	9783.17	8185.34
26	10136.9	10366.4	9260.18	70	10947	9754.86	8225.64
27	10404.6	10495.6	9771.44	71	11161.5	9724.47	8247.93
28	10718.5	10571.3	10151.5	72	13418.9	9958.9	9200.32
29	10975.5	10612.7	10015.2	73	11260.5	9970.43	9168.51
30	10326	10610.8	9371.86	74	10427.2	10405.6	9391.09
31	11111.1	10608.2	9493.73	75	10977.8	10553.5	10090.8
32	10136.4	10490.8	9807.78	76	11417.4	10728.3	10282.5
33	10370.5	10261.8	8625.07	77	10295.5	10674.2	9555.27
34	11152.5	10469.5	9694.3	78	11334.3	10765.8	9759.92
35	10115	10162.7	9059.34	79	10484	10602.2	9956.48
36	11421.3	10520.4	9826.67	80	10369.5	10400.4	8694.45
37	10741.9	10395.1	9979.28	81	11416.2	10359.2	8692.96
38	11516.5	10688	10011.5	82	9945.18	10103.3	8453.79
39	11238.7	10713.4	10392.8	83	10050.4	10014	8563.58
40	11041.5	10786.4	10200.9	84	12782.8	10077.4	9313.53
41	10428.2	10754.8	9984.06	85	10857.2	10139.3	9373.68
42	10389.1	10647.4	9290.48	86	10446.1	10415.4	9479.02
43	11649.1	10681.7	10334.3	87	10447.1	10521.3	9530.25
44	10517.8	10418.8	9697.4	88	11218	10658.5	10028.9

Lampiran 3.23. Contoh Perhitungan *Output*

Penentuan *output* akhir jaringan *recurrent neuro fuzzy* dapat diperoleh dengan persamaan sebagai berikut (Lin & Lee, 1996:510):

$$y_i^* = \frac{\sum_{s=1}^{r} \mu_{As}(x_i) f_s(x_i)}{\sum_{s=1}^{r} \mu_{As}(x_i)}; i = 1, 2, \dots, N \quad (3.53)$$

dengan,

s : banyak aturan *inferensi fuzzy, s*=1, 2,....*r*

y_i^* : *output* akhir jaringan.

$\mu_{As}(x_i)$: nilai keanggotaan tiap x_i dalam himpunan *fuzzy A* pada aturan *fuzzy* ke-*s* bagian anteseden

$f_s(x_i)$: *output* jaringan hasil pembelajaran tiap x_i pada setiap aturan inferensi *fuzzy* R^s pada bagian konsekuen

Nilai *output* data ke-1 (ribu)

	Cluster 1	*Cluster* 2	*Cluster* 3
$\mu_{As}(x_i)$	0,0115	-0,0354	1,00052
$f_s(x_i)$	11700,6	9480,1	7996,01

Sehingga nilai *output* ke-1 adalah

$$y_i^* = \frac{\sum_{s=1}^{r} \mu_{As}(x_i) f_s(x_i)}{\sum_{s=1}^{r} \mu_{As}(x_i)}$$

$$= \frac{(0{,}0115 \times 11700{,}6) + (-0{,}0354 \times 9480{,}1) + (1{,}00052 \times 7996{,}01)}{(0{,}0115) + (-0{,}0354) + (1{,}00052)}$$

$$= \frac{134{,}5569 - 335{,}59554 + 8000{,}1679}{0{,}97662} = \frac{7799{,}1293}{0{,}97662} = 7985{,}8382$$

Nilai *output* data ke-2 (ribu)

	Cluster 1	*Cluster* 2	*Cluster* 3
$\mu_{As}(x_i)$	-0,0333	0,09116	0,96635
$f_s(x_i)$	11298,6	9622,32	8303,01

Sehingga nilai *output* ke-2 adalah

$$y_i^* = \frac{\sum_{s=1}^{r} \mu_{As}(x_i) f_s(x_i)}{\sum_{s=1}^{r} \mu_{As}(x_i)}$$

$$= \frac{(-0{,}0333 \times 11298{,}6) + (0{,}09116 \times 9622{,}32) + (0{,}96635 \times 8303{,}01)}{(-0{,}0333) + (0{,}09116) + (0{,}96635)}$$

$$= \frac{-376{,}24338 + 877{,}17069 + 8023{,}6137}{1{,}02421} = \frac{8524{,}541}{1{,}02421} = 8323{,}04$$

Dan seterusnya

Lampiran 3.24 Bobot Hasil Pembelajaran

b_k =

[0.124 -2.488 0.995 -0.236 -0.871 -0.342 -1.189 0.697 1.603 2.535 1.690]

b_0 = -0.025

$w_{jk(b)}$ =

[-0.476 -0.013 -0.082 0.208 0.322 0.366 -0.362 0.280 0.211 -0.525 0.444;
-0.002 0.582 -0.242 0.121 -0.425 0.361 -0.396 0.009 0.271 0.608 0.535;
0.072 -0.458 -0.531 -0.344 0.525 -0.353 0.508 -0.399 0.583 -0.233 0.353;
-0.378 0.147 -0.040 -0.210 0.509 0.123 0.080 0.649 -0.291 0.400 0.296;
-0.182 0.086 -0.643 -0.633 0.047 0.402 0.701 -0.576 0.094 -0.048 0.569;
-0.247 -0.428 0.446 -0.268 0.044 -0.481 0.165 -0.369 0.209 0.294 0.289;
-0.075 -0.527 -0.411 0.586 -0.535 0.469 0.062 0.772 -0.573 -0.089 0.458;
0.702 -0.627 0.417 0.450 0.568 -0.598 -0.162 -0.374 0.408 -0.107 0.478;
-0.483 -0.299 -0.537 -0.516 0.569 0.115 0.081 -0.553 0.480 0.190 0.174;
0.020 -0.124 -0.642 -0.369 -0.580 -0.455 -0.420 -0.129 -0.612 0.626 0.518;
-0.014 -0.014 -0.246 0.567 -0.201 -0.559 0.453 -0.172 -0.351 -0.149 0.470]

$w_{ik(a)}$ =

[0.912 -1.272 -0.328 -0.303 -0.463 0.359 -0.965 0.828 -0.900 0.737 1.071;
1.007 2.936 1.789 -0.250 0.132 -2.262 1.185 -0.822 2.582 0.942 0.432;

-1.034 1.213 -1.663 0.522 -0.446 -1.956 1.833 -0.493 2.659 -1.739 0.606;
1.358 -2.403 -0.918 0.090 -1.265 1.933 -1.348 1.526 -2.578 -2.866 1.169]

v_k =

[-

-

2.483 2.666 1.688 0.942 -1.660 -3.005 -2.327 2.019 -2.433 -2.094 1.145]

Lampiran Bahasa Pemrograman Matlab untuk Proses Pembelajaran pada RNN dan NFS

X1=[]Bahasa Pemrograman Matlab untuk Proses Pembelajaran pada RNN dan NFS

1. Pembelajaran RNN dan mencari neuron NFS

```
X1clear all;clear;clc; %X1 dan X2 diganti sesuai
dengan matriks input-output yang digunakan
X1=[];%X1 adalah matriks input-output TRAINING
X2=[];%X2 adalah matriks input-output TESTING
P=[X1(:,1:7)']; %input TRAINING
Pc=[X2(:,1:7)']; %nput TESTING
[m, n]=size(P);
[mc, nc]=size(Pc);
T=X1(:,8)';%target output TRAINING
Tc=X2(:,8)';%target output Testing
%normalisasi input dan target output training dan
TESTING
[Pn,meanp,stdp,Tn,meant,stdt]=prestd(P,T);
[Pcn,meanpc,stdpc,Tcn,meantc,stdtc]=prestd(Pc,Tc);
%inisialisasi jaringan syaraf
n11=1; %jumlah neuron lapisan tersembunyi diubah-
ubah hingga neuron sesuai
net=newelm(minmax(Pn),[4 1],{'tansig' 'purelin'},
'traingdx');
BobotAwal_Input=net.IW{1,1};
BobotAwal_Bias_Input=net.b{1,1};
BobotAwal_Delay=net.LW{1,1};
BobotAwal_Lapisan=net.LW{2,1};
BobotAwal_Bias_Lapisan=net.b{2,1};
%set parameter
net.trainParam.epochs=5000;
net.trainParam.goal=1e-5;
net.trainParam.max_perf_inc=1.06;
net.trainParam.lr=0.1;
net.trainParam.lr_inc=1.2;
net.trainParam.lr_dec=0.6;
net.trainParam.mc=0.8;
net.trainParam.show=500;
%melakukan pebelajaran
net=train(net,Pn,Tn);
%melihat bobot akhir input, lapisan, dan bias
BobotAkhir_Input=net.IW{1,1};
BobotAkhir_Bias_Input=net.b{1,1};
BobotAkhir_Delay=net.LW{1,1};
```

```
BobotAkhir_Lapisan=net.LW{2,1};
BobotAkhir_Bias_Lapisan=net.b{2,1};
%menghitung MSE dan MAPE
ab = sim(net,Pn);
a=poststd(ab,meant,stdt)
e1 = T-a;
MSE = mse(e1)
mape1=[abs(((T-a)./T).*100)];
MAPE1=sum(mape1)/25
[m1,b1,r1] = postreg(a,T);
%menghitung MSE testing
Qn=trastd(Pc,meanp,stdp);
bn=sim(net,Qn)
b=poststd(bn,meant,stdt)
E2=Tc-b;
MSE2=mse(E2)
Mape2=[abs(((Tc-b)./Tc).*100)];
MAPE2=sum(Mape2)/6
```

2. Proses eliminasi variabel *input*

```
clear all;clear;clc;%X1 dan X2 diganti sesuai
matriks input-output yang digunakan
X1=[];%X1 adalah matriks input-output TRAINING
X2=[];%X2 adalah matriks input-output TESTING
P=[X1(:,1:6)']; %input TRAINING diganti sesuai input
yang dieliminasi
Pc=[X2(:,1:6)']; %input TESTING diganti sesuai input
yang dieliminasi
[m, n]=size(P);
[mc, nc]=size(Pc);
T=X1(:,8)';%target output TRAINING
Tc=X2(:,8)';%target output TESTING
%normalisasi input dan target output training dan
TESTING
[Pn,meanp,stdp,Tn,meant,stdt]=prestd(P,T);
[Pcn,meanpc,stdpc,Tcn,meantc,stdtc]=prestd(Pc,Tc);
%inisialisasi jaringan syaraf
n11=11; %jumlah neuron lapisan tersembunyi
net=newelm(minmax(Pn),[14 1],{'tansig' 'purelin'},
'traingdx');
BobotAwal_Input=net.IW{1,1};
BobotAwal_Bias_Input=net.b{1,1};
BobotAwal_Delay=net.LW{1,1};
BobotAwal_Lapisan=net.LW{2,1};
BobotAwal_Bias_Lapisan=net.b{2,1};
```

```
%set parameter
net.trainParam.epochs=5000;
net.trainParam.goal=1e-5;
net.trainParam.max_perf_inc=1.06;
net.trainParam.lr=0.1;
net.trainParam.lr_inc=1.2;
net.trainParam.lr_dec=0.6;
net.trainParam.mc=0.8;
net.trainParam.show=500;
%melakukan pebelajaran
net=train(net,Pn,Tn);
%melihat bobot akhir input, lapisan, dan bias
BobotAkhir_Input=net.IW{1,1};
BobotAkhir_Bias_Input=net.b{1,1};
BobotAkhir_Delay=net.LW{1,1};
BobotAkhir_Lapisan=net.LW{2,1};
BobotAkhir_Bias_Lapisan=net.b{2,1};
%menghitung MSE
ab = sim(net,Pn);
a=poststd(ab,meant,stdt)
e1 = T-a;
MSE = mse(e1)
mape1=[abs(((T-a)./T).*100)];
MAPE1=sum(mape1)/55
[m1,b1,r1] = postreg(a,T);
%normalisai input dan target output TESTING
Qn=trastd(Pc,meanp,stdp);
bn=sim(net,Qn)
b=poststd(bn,meant,stdt)
E2=Tc-b;
MSE2=mse(E2)
Mape2=[abs(((Tc-b)./Tc).*100)];
MAPE2=sum(Mape2)/19
```

3. Proses *clustering*

```
clear all;clear;clc; %X1 dan X2 adalah matriks
input-output sesuai hasil eliminasi
X1=[];%X1 adalah input-output TRAINING
X2=[];%X2 adalah input-output TESTING
P=[X1(:,1:6)']; %input TRAINING
Pc=[X2(:,1:6)']; %input TESTING
[m, n]=size(P);
[mc, nc]=size(Pc);
T=X1(:,8)';%target output TRAINING
Tc=X2(:,8)';%target output TRAINING
```

```
%normalisasi input dan target output training dan
TESTING
[Pn,meanp,stdp,Tn,meant,stdt]=prestd(P,T);
[Pcn,meanpc,stdpc,Tcn,meantc,stdtc]=prestd(Pc,Tc);
n12=18; %jumlah neuron lapisan tersembunyi
net=newelm(minmax(Pn),[14 1],{'tansig' 'purelin'},
'traingdx');
BobotAwal_Input=net.IW{1,1};
BobotAwal_Bias_Input=net.b{1,1};
BobotAwal_Delay=net.LW{1,1};
BobotAwal_Lapisan=net.LW{2,1};
BobotAwal_Bias_Lapisan=net.b{2,1};
%set parameter
net.trainParam.epochs=5000;
net.trainParam.goal=1e-5;
net.trainParam.max_perf_inc=1.06;
net.trainParam.lr=0.1;
net.trainParam.lr_inc=1.2;
net.trainParam.lr_dec=0.6;
net.trainParam.mc=0.8;
net.trainParam.show=500;
%melakukan pebelajaran
net=train(net,Pn,Tn);
%melihat bobot akhir input, lapisan, dan bias
BobotAkhir_Input=net.IW{1,1};
BobotAkhir_Bias_Input=net.b{1,1};
BobotAkhir_Delay=net.LW{1,1};
BobotAkhir_Lapisan=net.LW{2,1};
BobotAkhir_Bias_Lapisan=net.b{2,1};
%menghitung MSE
ab = sim(net,Pn);
E1 = Tn-ab;
MSE = mse(E1)
%melakukan simulasi
PPn = [Pn Pcn];
TTn = [Tn Tcn];
y_1 = sim(net,PPn);
SSE = sum((y_1-TTn).^2);
[m0,b,r] = postreg(y_1(1,:),TTn);
%clustering dengan FCM
X3=[P; T]; %TRAINING yang akan dicluster
X4=[Pc; Tc]; %TESTING yang akan dicluster
X3=X3';
X4=X4';
C=3; %jumlah cluster disesuaikan
[V,U,obj_fcn]=fcm(X3,C)
[Vc,Uc,obj_fcn_c]=fcm(X4,C)
```

```
[DT, II]=max(U)
[DTc, IIc]=max(Uc)
for k=1:C,
for j=1:n,
if II(j)==k,
TA(j,k)=1;
else
TA(j,k)=0;
end;
end;
end;
for k=1:C,
for j=1:nc,
if IIc(j)==k,
TAc(j,k)=1;
else
TAc(j,k)=0;
end;
end;
end;
%menghitung nilai keanggotaan tiap data pada bagian
anteseden
n12=18; %jumlah neuron lapisan tersembunyi
%inisialisasi jaringan syaraf
net=newelm(minmax(Pn),[14   3],{'tansig'   'purelin'
'traingdx'});
BobotAwal_Input=net.IW{1,1};
BobotAwal_Bias_Input=net.b{1,1};
BobotAwal_Delay=net.LW{1,1};
BobotAwal_Lapisan=net.LW{2,1};
BobotAwal_Bias_Lapisan=net.b{2,1};
%set parameter
net.trainParam.epochs=5000;
net.trainParam.goal=1e-5;
net.trainParam.max_perf_inc=1.06;
net.trainParam.lr=0.1;
net.trainParam.lr_inc=1.2;
net.trainParam.lr_dec=0.6;
net.trainParam.mc=0.8;
net.trainParam.show=500;
%melakukan pembelajaran
net=train(net,Pn, TA');
%melihat bobot akhir input, lapisan, dan bias
BobotAkhir_Input=net.IW{1,1};
BobotAkhir_Bias_Input=net.b{1,1};
BobotAkhir_Delay=net.LW{1,1};
BobotAkhir_Lapisan=net.LW{2,1};
```

```
BobotAkhir_Bias_Lapisan=net.b{2,1};
%menghitung nilai MSE
y2=sim(net,Pn);
E2=TA'-y2;
perf_2=mse(e2)
%melakukan simulasi
muA=sim(net,Pn)  %nilai  keanggotaan  tarining  data
pada bagian anteseden
muAc=sim(net,Pcn)  %nilai  keanggotaan  TESTING  pada
bagian anteseden
```

4. Penyederhanaan bagian konsekuen (bagian *THEN*) atau mencari parameter dengan LSE

```
function T=LSE(A,y);
[n m]=size(A);
n1=m;
n2=n-n1;
A1=A(1:n1,:);
y1=y(1:n1,:);
A2=A(n1+1:end,:);
y2=y(n1+1:end,:);
P=inv(A1'*A1);
T=P*A1'*y1;
for i=1:n2,
    P=P-
(P*A2(i,:)'*A2(i,:)*P)/(1+A2(i,:)*P*A2(i,:)');
    T=T+P*A2(i,:)'*(y2(i,:)-A2(i,:)*T);
end;
 D=A*T;
k=1:n;

E1=[];%E1  adalah  TRAINING  setiap  cluster  setelah
proses eliminasi
R1=LSE(E1(:,1:6),E1(:,7));%yr1=E1(:,1:6)*R1;
E=E1(:,7)-yr1;
e=mean(E);
R1=[R1' e]
```

5. Pembelajaran jaringan syaraf bagian konsekuen (bagian *THEN*) untuk setiap RNN_s

```
clear all;clear;clc;
X1=[];% matriks input-output TRAINING awal
X2=[];%matriks input-output TESTING awal
P=[X1(:,1:6)'];%input TRAINING
Pc=[X2(:,1:6)'];%input TESTING
```

```
[m,n]=size(P);
[mc,nc]=size(Pc);
T=X1(:,7)';%target output TRAINING
Tc=X2(:,7)';%target output TESTING
[Pn,meanp,stdp,Tn,meant,stdt]=prestd(P,T);
[Pcn,meanpc,stdpc,Tcn,meantc,stdtc]=prestd(Pc,Tc);
E1=[];%matriks TRAINING setiap kelas
E2=[];%matriks TESTING setiap kelas
P1=[E1(:,1:6)']; % input TRAINING RNNs
[m1,n1]=size(P1);
Pc1=E2(:,1:6)'; %input TESTING RNNs
[mc1,nc1]=size(Pc1);
T1=E1(:,7)';%target output TRAINING RNNs
Tc1=E2(:,7)';%target output TESTING RNNs
%normalisasi THD & TESTING
[p1n,meanp1,stdp1,t1n,meant1,stdt1]=prestd(P1,T1);
[pc1n,meanpc1,stdpc1,tcln,meantc1,stdtc1]=prestd(Pc1
,Tc1);
[p2n,meanp2,stdp2,t2n,meant2,stdt2]=prestd(P,T);
[pc2n,meanpc2,stdpc2,tc2n,meantc2,stdtc2]=prestd(Pc,
Tc);
%inisialisasi
net=newelm(minmax(Pn),[4    1],{'tansig'    'purelin'
'traingdx'});
BobotAwal_Input=net.IW{1,1};
BobotAwal_Bias_Input=net.b{1,1};
BobotAwal_Delay=net.LW{1,1};
BobotAwal_Lapisan=net.LW{2,1};
BobotAwal_Bias_Lapisan=net.b{2,1};
%set parameter
net.trainParam.epochs=5000;
net.trainParam.goal=1e-5;
net.trainParam.max_perf_inc=1.06;
net.trainParam.lr=0.1;
net.trainParam.lr_inc=1.2;
net.trainParam.lr_dec=0.6;
net.trainParam.mc=0.8;
net.trainParam.show=500;
%melakukan pembelajaran
 net=train(net,p1n,t1n);
%melihat bobot akhir input, lapisan, dan bias
BobotAkhir_Input=net.IW{1,1};
BobotAkhir_Bias_Input=net.b{1,1};
BobotAkhir_Delay=net.LW{1,1};
BobotAkhir_Lapisan=net.LW{2,1};
BobotAkhir_Bias_Lapisan=net.b{2,1};
%melakukan simulasi
```

```
muS01=sim(net,p2n);
muS1=sim(net,pc2n);
muS0a=poststd(muS01,meant1,stdt1);
 muSa=poststd(muS1,meant1,stdt1);
 muAS=[];
muAA=[];
muAS=[muAS;muSa]
muAA=[muAA;muS0a]
```

6. Penentuan Output Akhir

```
clear all;clear;clc;% data eliminasi pertama
X1=[];% matriks input-output TRAINING awal
X2=[];% matriks input-output TESTING awal
P=[X1(:,1:7)']; %input TRAINING
Pc=[X2(:,1:7)']; %input TESTING
[m, n]=size(P);
[mc, nc]=size(Pc);
T=X1(:,8)';%target output TRAINING
muA=[];%matriks TRAINING hasil pembelajaran jaringan
syaraf bagian IF
muAc=[];%matriks TESTING hasil pembelajaran jaringan
syaraf bagian IF
muAA=[];%matriks    TRAINING    hasil    pembelajaran
jaringan syaraf bagian THEN
muAS=[];%matriks TESTING hasil pembelajaran jaringan
syaraf bagian THEN
muA=muA';
muAc=muAc';
muAA=muAA';
muAS=muAS';
%menghitung output jaringan untuk TRAINING
for i=1:55,
yt0(i)=0;
st0(i)=0;
for k=1:3,
yt0(i)=yt0(i)+muA(k,i)*muAA(k,i);
st0(i)=st0(i)+muA(k,i);
end;
yt0(i)=yt0(i)/st0(i)
end;
%menghitung output jaringan untuk TESTING
for i=1:19,
yt1(i)=0;
st1(i)=0;
for k=1:3,
yt1(i)=yt1(i)+muAc(k,i)*muAS(k,i);
```

```
st1(i)=st1(i)+muAc(k,i);
end;
yt1(i)=yt1(i)/st1(i)
end;
E=T-yt0;
mse_TRAINING=mse(E)
Ec=Tc-yt1;
mse_TESTING=mse(Ec)
mape1=[abs(((T-yt0)./T).*100)];
MAPE1=sum(mape1)/55
mape2=[abs(((Tc-yt1)./Tc).*100)];
MAPE2=sum(mape2)/19
```