

**Pengenalan Pembicara dengan
Jaringan Syaraf Tiruan *BACKPROPAGATION***

BASKORO OKTIANTO



**DEPARTEMEN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT PERTANIAN BOGOR
BOGOR
2004**

ABSTRAK

BASKORO OKTIANTO. Pengenalan Pembicara dengan Jaringan Syaraf Tiruan *Backpropagation* (*Speaker Recognition with Backpropagation Artificial Neural Network*). Dibimbing oleh **SUGI GURITMAN** dan **AHMAD RIDHA**.

Masalah pengenalan pembicara terbagi menjadi dua bagian, yaitu identifikasi pembicara (menentukan identitas pembicara) dan verifikasi pembicara (melakukan verifikasi identitas yang diklaim oleh pembicara). Pengenalan pembicara termasuk dalam masalah *nonalgorithmic* maka dapat digunakan jaringan syaraf tiruan (JST) untuk mengenali pola-pola suara pembicara.

Dalam penelitian ini dibangun suatu sistem yang dapat melakukan pengenalan pembicara menggunakan JST *backpropagation*. Sebelum diproses dalam JST data suara terlebih dahulu diproses dengan proses-proses sinyal digital melalui suatu proses *feature extraction* ditambah dengan proses *feature selection*. Proses *feature extraction* dilakukan dengan analisis *cepstral* dan *feature selection* dilakukan dengan *principal component analysis*. Hasil dari JST selanjutnya diolah oleh model pembuatan keputusan. Model pembuatan keputusan dalam sistem identifikasi akan menentukan identitas pembicara dan dalam sistem verifikasi akan menerima atau menolak klaim yang diajukan oleh pembicara.

Sistem pengenalan pembicara yang dibangun mampu mengidentifikasi dengan tingkat generalisasi tertinggi sebesar 92,3077% dan melakukan verifikasi dengan nilai *equal error rate* sebesar 6,5657%.

**PENGENALAN PEMBICARA DENGAN
JARINGAN SYARAF TIRUAN *BACKPROPAGATION***


BASKORO OKTIANTO

**Skripsi
Sebagai salah satu syarat untuk memperoleh gelar
Sarjana Komputer
pada
Departemen Ilmu Komputer**


**DEPARTEMEN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT PERTANIAN BOGOR
BOGOR
2004**

Judul Skripsi : Pengenalan Pembicara dengan Jaringan Syaraf Tiruan *Backpropagation*
Nama : Baskoro Oktianto
NRP : G06499043
Departemen : Ilmu Komputer

Menyetujui,

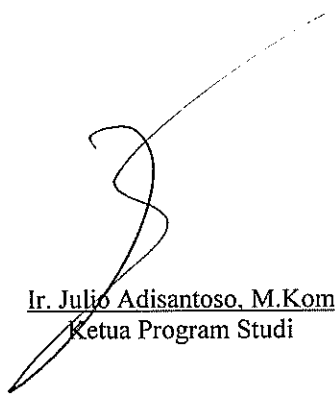


Dr. Sugi Guritman
Pembimbing I

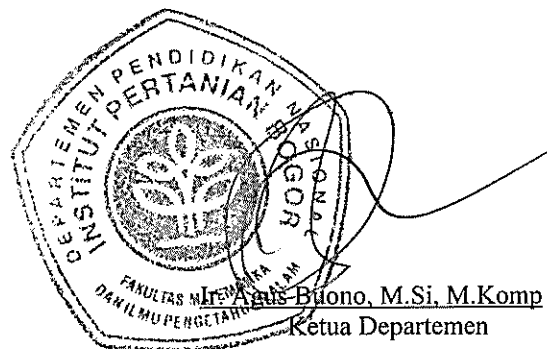


Ahmad Ridha, S.Kom
Pembimbing II

Mengetahui,



Ir. Julio Adisantoso, M.Komp
Ketua Program Studi



Ir. Agus Buono, M.Si, M.Komp
Ketua Departemen

RIWAYAT HIDUP

Penulis dilahirkan di Jakarta pada tanggal 14 Oktober 1980 dari orang tua yang bernama Darso Harsono dan Sarmini. Penulis merupakan anak kedua dari dua bersaudara.

Tahun 1999 penulis lulus dari SMU Negeri 12 Jakarta dan pada tahun yang sama melanjutkan studi ke Institut Pertanian Bogor melalui jalur Ujian Masuk Perguruan Tinggi Negeri. Penulis memilih Program Studi Ilmu Komputer, Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam.

Penulis pernah melakukan kerja praktik di PUSLIT LIPI Biologi Bogor pada bulan Januari sampai dengan bulan Maret 2003.

PRAKATA

Puji dan syukur penulis panjatkan kepada Allah *subhanahu wa ta'ala* atas rahmat dan petunjuk-Nya sehingga penulis dapat menyelesaikan tugas akhir ini. Tema yang dipilih dalam tugas akhir ini adalah sistem pengenalan pembicara menggunakan jaringan syaraf tiruan yang melibatkan bidang pemrosesan sinyal digital dan berjudul Pengenalan Pembicara dengan Jaringan Syaraf Tiruan *Backpropagation*.

Penulis mengucapkan terima kasih kepada Bapak Dr. Sugi Guritman dan Bapak Ahmad Ridha, S.Kom yang telah memberikan saran, koreksi, dan bimbingan selama pengerjaan tugas akhir ini. Selanjutnya penulis juga mengucapkan terima kasih kepada:

1. Kedua orang tua dan kakak penulis atas doa dan dukungan yang selalu diberikan.
 2. Rekan-rekan sepenelitian, khususnya Hanief, Diana, Muhidin, dan Ono atas diskusi dan bahan-bahannya yang sangat bermanfaat.
 3. Seluruh rekan-rekan yang telah bersedia diambil contoh suaranya untuk digunakan dalam pengerjaan tugas akhir.
 4. Mujiburrohman, Asep Purnomo Yudo, Elia Natari Barito, dan Sofareja Muhtar yang tergabung dalam keluarga besar MABES atas keceriaan yang diberikan.
 5. Likha, Danny, Ade, Rakhmat, Nina, dan Copit atas bantuannya.
 6. Dewis, Ary, Meli, dan Siska atas bantuan dan saran-sarannya terutama dalam masalah penulisan.
 7. Seluruh *Ilkomerz* atas dukungan dan kebersamaannya.
 8. Seluruh staf pengajar dan pegawai Departemen Ilmu Komputer.
 9. Pihak-pihak lain yang tidak dapat disebutkan satu per satu.
- Semoga tugas akhir ini dapat bermanfaat.

Bogor, Juni 2004

BASKORO OKTIANTO

DAFTAR ISI

	Halaman
DAFTAR TABEL.....	vi
DAFTAR GAMBAR.....	vi
DAFTAR LAMPIRAN.....	vi
PENDAHULUAN	1
Latar Belakang	1
Tujuan	1
Ruang Lingkup.....	1
Output dan Manfaat.....	1
TINJAUAN PUSTAKA	1
Pengenalan Pembicara	1
Akuisisi Data Suara Digital.....	2
<i>Feature Extraction</i>	3
<i>Feature Selection</i>	5
Pembentukan Model Referensi Pembicara dan Pencocokan Pola.....	7
Pembuatan Keputusan.....	9
Pengujian Pengenalan Pembicara.....	10
METODE PENELITIAN.....	11
Langkah-langkah Pengembangan Sistem.....	11
Data Teknis	12
Spesifikasi Perangkat Keras dan Lunak	13
Pengujian Sistem.....	13
HASIL DAN PEMBAHASAN.....	13
Akuisisi Data Suara Digital.....	13
<i>Feature Extraction</i> dan <i>Feature Selection</i>	13
Pembentukan Model Referensi Pembicara dan Pencocokan Pola serta Hasil Identifikasi (Pembelajaran dan Generalisasi JST).....	14
Verifikasi.....	15
Kelebihan dan Keterbatasan Sistem.....	15
KESIMPULAN DAN SARAN.....	16
Kesimpulan	16
Saran	16
DAFTAR PUSTAKA	17
LAMPIRAN.....	18

DAFTAR TABEL

	Halaman
1. Data proses sinyal digital	12
2. Data JST.....	12
3. Contoh definisi target JST untuk 13 pembicara	12
4. Hasil percobaan pertama identifikasi pembicara.....	14

DAFTAR GAMBAR

	Halaman
1. Tahapan pengenalan pembicara.	2
2. Contoh grafik sinyal digital.....	3
3. <i>Frame blocking</i> pada sinyal suara.	3
4. Grafik <i>hamming window</i> dengan N sama dengan 256.	4
5. Grafik analisis <i>cepstral</i> dibandingkan dengan sinyal asli.	4
6. Contoh grafik akar ciri yang telah diurutkan.	7
7. Model JST.....	7
8. Model JST <i>backpropagation</i>	8
9. Grafik akar ciri data pembelajaran.....	13
10. Grafik <i>detection error tradeoff</i> (DET)	15

DAFTAR LAMPIRAN

	Halaman
1. Algoritme pembelajaran JST <i>backpropagation</i>	19
2. Tabel percobaan jumlah neuron tersembunyi dengan tiga kali pengulangan pada toleransi galat 0,001 dan laju pembelajaran 0,1	20
3. Tabel hasil pengujian identifikasi	21
4. Tabel hasil identifikasi secara terperinci.....	22
5. Contoh tampilan program	24

PENDAHULUAN

Latar Belakang

Identitas memegang peranan yang sangat penting dalam kehidupan manusia. Proses identifikasi atau verifikasi banyak digunakan dalam kehidupan sehari-hari, misalnya dalam penggunaan mesin ATM atau otorisasi seseorang untuk memasuki suatu wilayah tertentu.

Proses identifikasi atau verifikasi umumnya dilakukan dengan suatu alat identifikasi seperti kartu ATM atau kartu khusus tertentu. Bila kartu tersebut hilang tentunya akan menjadi masalah bagi pemiliknya. Dengan semakin berkembangnya bidang pengenalan pola sekarang ini, proses identifikasi atau verifikasi dapat dilakukan secara biometrik. Dengan teknik biometrik proses identifikasi atau verifikasi dapat dilakukan melalui karakteristik fisiologi atau perilaku seseorang (Xafopoulos, 2001). Beberapa cara untuk melakukan identifikasi atau verifikasi secara biometrik adalah melalui suara, wajah, sidik jari, tanda tangan, retina dan lain-lain. Beberapa hal yang mendorong penggunaan identifikasi atau verifikasi secara biometrik adalah biometrik bersifat universal (terdapat pada setiap orang), unik (tiap orang mempunyai ciri khas tersendiri), dan tidak mudah dipalsukan (Xafopoulos, 2001). Dengan teknik biometrik seseorang tidak harus membawa suatu alat identifikasi seperti pada teknik konvensional.

Proses identifikasi atau verifikasi dengan suara memiliki keunggulan dibandingkan dengan karakteristik yang lain. Identifikasi atau verifikasi dengan suara hanya membutuhkan alat tambahan berupa mikrofon dan kartu suara sedangkan karakteristik-karakteristik yang lain misalnya sidik jari atau wajah membutuhkan alat tambahan seperti *scanner*. Hal ini sedikit banyak dapat menekan biaya pengembangan sistem.

Identifikasi atau verifikasi melalui suara termasuk dalam masalah *nonalgorithmic*. Walaupun sirkuit digital (komputer) mempunyai kecepatan yang jauh lebih tinggi daripada otak manusia tetapi dalam memproses masalah-masalah *nonalgorithmic* otak manusia lebih unggul (Fu, 1994). Suatu teknik yang dibuat dengan memodelkan otak manusia adalah jaringan syaraf tiruan (JST) atau *artificial neural network*. Seperti pada otak manusia, JST terdiri atas neuron-neuron yang saling berhubungan yang dapat bekerja sama satu dengan yang lainnya untuk membentuk suatu sistem. JST dapat belajar

untuk mengenali suatu pola melalui pembelajaran dan diharapkan dapat memecahkan masalah-masalah yang bersifat *nonalgorithmic*.

Tujuan

Penelitian ini bertujuan untuk mengembangkan suatu sistem yang dapat mengenali seseorang melalui kata-kata yang diucapkan oleh orang tersebut menggunakan teknik-teknik proses sinyal digital (*frame blocking* dan *frame windowing*), analisis *cepstral* untuk *feature extraction*, analisis komponen utama untuk *feature selection* dan JST *backpropagation* untuk pencocokan pola.

Ruang Lingkup

Ruang lingkup penelitian dibatasi pada pengenalan pembicara melalui kata-kata yang diucapkan oleh pembicara tersebut. Kata-kata yang diucapkan bersifat *text-dependent* yang berarti telah disepakati sebelumnya dan akan digunakan seterusnya. Akuisisi data dilakukan dengan mikrofon. Pengenalan dilakukan menggunakan JST dengan arsitektur *multi-layer perceptron* dan pembelajaran *backpropagation*. Sebelum di-input-kan dalam JST, data hasil akuisisi terlebih dahulu diproses dengan proses-proses sinyal digital (*frame blocking* dan *frame windowing*), kemudian dilakukan *feature extraction* menggunakan analisis *cepstral* dan *feature selection* menggunakan analisis komponen utama.

Output dan Manfaat

Sistem pengenalan pembicara ini dapat digunakan untuk melakukan identifikasi atau verifikasi seseorang melalui kata-kata yang diucapkan oleh orang tersebut. Pada identifikasi sistem memberikan *output* berupa identitas pengguna sistem, sedangkan pada verifikasi sistem memberikan pernyataan bahwa verifikasi diterima atau ditolak.

Sistem pengenalan pembicara antara lain bermanfaat untuk melakukan identifikasi, sebagai semacam *password* atau sebagai aplikasi absensi.

TINJAUAN PUSTAKA

Pengenalan Pembicara

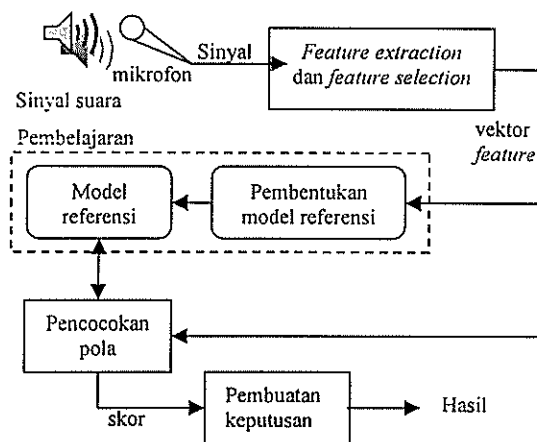
Menurut Owens (1993), pengenalan pembicara terbagi menjadi dua bagian, yaitu identifikasi pembicara dan verifikasi pembicara.

Identifikasi pembicara adalah proses untuk menentukan identitas suara pembicara, sedangkan verifikasi pembicara berfungsi untuk memverifikasi kesesuaian suara pembicara dengan identitas yang diklaim oleh pembicara. Pengenalan pembicara lebih menitikberatkan pada pengenalan suara pembicara dan tidak pada pengenalan ucapan pembicara.

Menurut Campbell (1997) secara umum sistem pengenalan pembicara mempunyai lima tahapan sebagai berikut dengan diagram bloknya diilustrasikan pada Gambar 1:

1. **Akuisisi data suara digital.** Proses untuk mengakuisisi ucapan pembicara dan mengubahnya menjadi sinyal digital. Sinyal digital yang terbentuk berupa suatu vektor yang merepresentasikan suara pembicara.
2. **Feature extraction dan feature selection.** *Feature extraction* mengekstrak data hasil akuisisi sehingga dihasilkan data yang berdimensi lebih kecil. Data hasil *feature extraction* kemudian dimasukkan dalam proses *feature selection* yang mengubah dari ruang data ke ruang *feature* yang berdimensi kecil.
3. **Pembentukan model referensi pembicara.** Pembentukan model referensi pembicara merupakan tahapan pembelajaran dan akan membentuk suatu model referensi agar sistem dapat mengenali pembicara. Tahap ini memerlukan data berupa vektor-vektor *feature* hasil dari *feature extraction* dan *feature selection* yang mencakup seluruh pembicara. Model referensi yang terbentuk akan digunakan dalam pencocokan pola. Pembentukan model referensi pembicara merupakan tahapan khusus yang dilakukan pada waktu awal sebelum sistem siap digunakan. Tahap ini hanya dilakukan satu kali dan setelah dilakukan maka sistem siap untuk digunakan.
4. **Pencocokan pola (pattern matching).** Proses pencocokan pola menerima data yang telah diolah dengan *feature extraction* dan *feature selection* sebagai data *input*. Proses pencocokan pola akan mencocokkan pola data *input* dengan model referensi dan memberikan hasil berupa besarnya skor kesesuaian data *input* dengan pola-pola referensi yang ada.
5. **Pembuatan keputusan.** Proses pembuatan keputusan meliputi pembuatan keputusan untuk identifikasi dan untuk verifikasi. Pembuatan keputusan akan menerima skor hasil pencocokan pola. Pada sistem identifikasi, pembuatan keputusan akan

menentukan identitas pembicara dan pada sistem verifikasi akan menerima atau menolak pembicara.



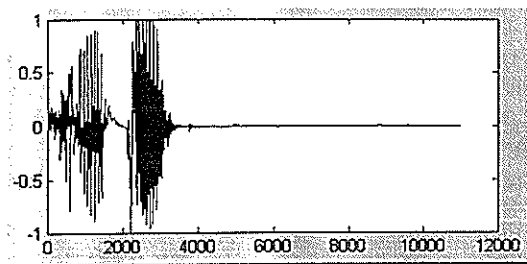
Gambar 1. Tahapan pengenalan pembicara.

Akuisisi Data Suara Digital

Suara merupakan gelombang analog yang dapat ditangkap oleh mikrofon. Sinyal analog tersebut dapat diubah menjadi sinyal digital melalui proses *sampling*, yaitu proses untuk memperoleh nilai dari sinyal analog dalam waktu diskret (Owens, 1993). Proses *sampling* menghasilkan suatu vektor berisi deretan bilangan yang merupakan representasi digital dari sinyal suara. Hal yang perlu diperhatikan dalam melakukan *sampling* adalah frekuensi *sampling* (f_s), yaitu jumlah *sample* dalam 1 detik. Besaran f_s akan menentukan waktu diskret untuk melakukan *sampling*. Semakin besar f_s maka semakin besar ukuran data yang diperoleh dengan kualitas suara yang semakin baik, sedangkan semakin kecil f_s maka ukuran data yang diperoleh akan semakin kecil dengan konsekuensi penurunan kualitas suara. Umumnya f_s yang digunakan berkisar pada rentang 6-20 kHz (Owens, 1993). Jika dilakukan proses *sampling* selama t detik dengan $f_s = f$ maka akan diperoleh suatu vektor s yang mempunyai elemen sebanyak $f \times t$ sebagai berikut:

$$s^T = [s_1 \ s_2 \ s_3 \ \dots \ s_n]$$

Vektor s merepresentasikan sinyal suara dan dapat disebut sebagai sinyal suara digital. Proses selanjutnya akan menggunakan vektor s sebagai data *input*. Contoh grafik sinyal suara digital yang dibentuk dari vektor s diberikan dalam Gambar 2.



Gambar 2. Contoh grafik sinyal suara digital.

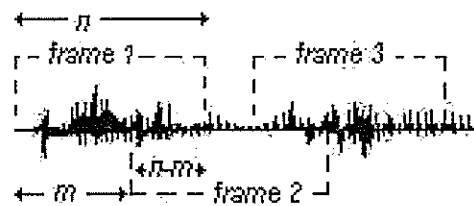
Feature Extraction

Tujuan dari *feature extraction* adalah untuk mengubah sinyal suara digital menjadi suatu representasi data yang berdimensi lebih kecil untuk diproses lebih lanjut. Manfaat yang diperoleh dari *feature extraction* adalah memudahkan dan mempercepat proses-proses selanjutnya. Hal ini dapat dilakukan karena *feature extraction* dapat mengekstrak informasi yang terdapat dalam sinyal suara digital.

Sebelum dilakukan ekstraksi ciri terlebih dahulu dilakukan langkah-langkah yang terdiri atas (1) *frame blocking*, dan (2) *frame windowing* (Xafopoulos, 2001).

1. **Frame blocking.** Dalam analisis sinyal digital terdapat suatu konsep yang dinamakan *short-time analysis* (Owens, 1993). Asumsi yang digunakan adalah dalam interval waktu yang panjang, pola gelombang suara tidak stasioner, tetapi dalam waktu yang cukup pendek (10-30 milidetik) dapat dikatakan stasioner. Hal ini dikarenakan kecepatan perubahan spektrum suara berkaitan dengan kecepatan perubahan organ-organ penghasil suara pada manusia dan hal ini dibatasi oleh keterbatasan fisiologi. Berdasarkan pada hal di atas, sinyal suara digital yang telah diakuisisi dapat dibagi-bagi menjadi segmen-segmen dengan durasi 10-30 milidetik. Segmen sinyal suara digital ini disebut dengan *frame* dan proses pembentukan *frame-frame* disebut dengan *frame blocking* dengan tiap *frame* direpresentasikan dalam sebuah vektor.

Dalam pembentukan *frame* umumnya terdapat *overlap* antara *frame-frame* yang bersebelahan. Jika panjang *frame* adalah n , maka pada tiap-tiap *frame* akan terdapat *overlap* sebesar $n - m$ dengan $m < n$. Contoh ilustrasi *frame blocking* dalam bentuk grafik dapat dilihat pada Gambar 3.



Gambar 3. Frame blocking pada sinyal suara.

Dalam representasi numerik misalnya terdapat suatu vektor sinyal suara digital s dengan frekuensi *sampling* f_s dan durasi perekaman t :

$$s^T = [s_1 \ s_2 \ s_3 \ \dots \ s_{ft}]$$

Jika didefinisikan panjang *frame* adalah n maka *frame* pertama adalah suatu vektor dengan elemen dimulai dari s_1 sampai s_n . Jika didefinisikan suatu besaran m maka *frame* kedua akan dimulai dari s_{m+1} sampai s_{m+n} sehingga antara *frame* pertama dan *frame* kedua akan terdapat *overlap* sebanyak $n - m$ elemen. *Frame* ketiga akan dimulai dari s_{2m+1} sampai s_{2m+n} , sehingga antara *frame* kedua dan ketiga akan terdapat *overlap* sebanyak $(m+n) - (2m)$ atau juga $n - m$. Hal ini terus dilakukan sampai seluruh vektor s telah tersegmentasi dalam *frame-frame*. Jika ternyata dalam *frame* terakhir jumlah elemennya kurang dari n maka *frame* tersebut dapat diabaikan karena umumnya sinyal suara pada bagian akhir tidak mengandung informasi yang penting.

Hasil dari *frame blocking* adalah suatu matriks yang tiap kolomnya adalah tiap *frame* dan diposisikan terurut dengan *frame* pertama menempati kolom pertama, *frame* kedua menempati kolom kedua dan seterusnya. Misalnya diberikan matriks F hasil *frame blocking* dengan banyaknya *frame* adalah b maka matriks tersebut dapat diilustrasikan sebagai berikut:

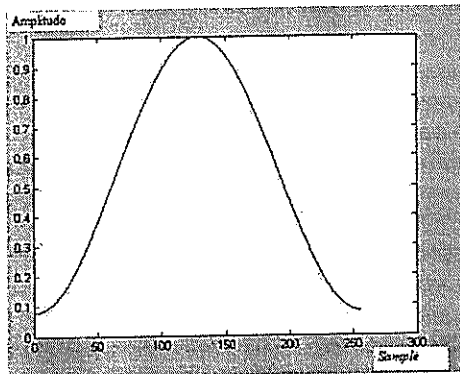
$$F = [frame_1 \ frame_2 \ frame_3 \ \dots \ frame_b]$$

2. **Frame windowing.** Proses *frame blocking* menyebabkan terjadinya *spectral leakage*, yaitu distorsi frekuensi pada bagian tepi *frame* yang dipengaruhi oleh *frame* di sebelahnya. *Frame windowing* bertujuan untuk meminimalkan diskontinuitas sinyal atau *spectral leakage* pada bagian awal dan akhir pada tiap *frame* (Xafopoulos, 2001). Metode untuk melakukan *frame windowing* adalah dengan memboboti (mengalikan) tiap *frame*

dengan suatu *window*. *Window* yang biasa digunakan contohnya adalah *hamming window*. *Hamming window* didefinisikan dalam persamaan berikut:

$$w[k] = 0,54 - 0,64 \cos\left(\frac{2\pi k}{N-1}\right), 0 \leq k \leq N-1$$

dengan N adalah panjang *window*. Jika diilustrasikan *hamming window* akan tampak seperti pada Gambar 4.



Gambar 4. Grafik *hamming window* dengan N sama dengan 256.

Hamming window adalah sebuah vektor dengan jumlah elemen sebanyak N . Besarnya N akan disesuaikan dengan banyaknya elemen pada *frame* yang akan diboboti sehingga banyaknya elemen pada *hamming window* akan sama dengan banyaknya elemen pada *frame* yang akan diboboti. Jika diberikan suatu vektor fr yang merupakan sebuah *frame* dan vektor w adalah *hamming window* maka pembobotan dilakukan dengan mengalikan elemen pertama pada fr dengan elemen pertama pada w , kemudian dilanjutkan dengan mengalikan elemen kedua pada fr dengan elemen kedua pada w dan seterusnya sampai elemen yang terakhir sehingga akan terbentuk vektor baru yang merupakan *frame* yang telah diboboti dengan *hamming window*. Jika diberikan vektor h yang merupakan hasil dari *frame windowing* untuk satu *frame* maka dapat diilustrasikan proses pembobotan sebagai berikut:

$$fr^T = [fr_1 \ fr_2 \ fr_3 \ \dots \ fr_n]$$

$$w^T = [w_1 \ w_2 \ w_3 \ \dots \ w_N]$$

$$h^T = [fr_1 w_1 \ fr_2 w_2 \ fr_3 w_3 \ \dots \ fr_n w_N]$$

Setelah pembobotan selesai dilakukan maka akan dihasilkan suatu matriks baru yang

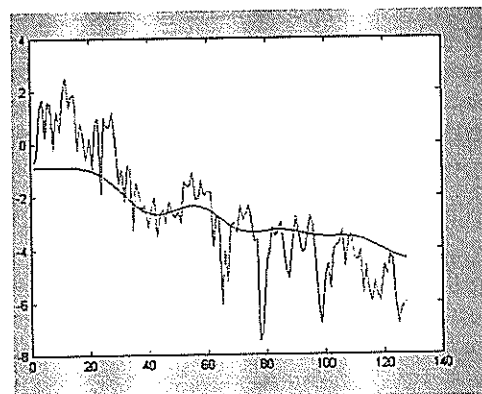
merupakan matriks hasil *frame blocking* yang telah terboboti. Jika diberikan matriks Z yang merupakan hasil dari *frame windowing* dan h_i adalah *frame* yang telah terboboti untuk *frame* ke- i dan b adalah banyaknya *frame*, maka matriks Z dapat diilustrasikan sebagai berikut:

$$Z = [h_1 \ h_2 \ h_3 \ \dots \ h_b]$$

Setelah *frame blocking* dan *frame windowing* selesai dilakukan maka pada tiap-tiap *frame* dilakukan *feature extraction*. Salah satu metode *feature extraction* yang sering digunakan dalam analisis sinyal digital adalah analisis *cepstral*. Analisis *cepstral* dapat mereduksi sebuah *frame*. Jika sebuah *frame* diberikan oleh vektor h maka nilai dari vektor *cepstral* diberikan oleh persamaan berikut:

$$\text{vektor cepstral} = \text{real}(\text{ifft}(\log(\text{abs}(\text{fft}(h))))))$$

dengan fft adalah transformasi fourier dan ifft adalah invers dari transformasi fourier. Dari vektor *cepstral* dapat diambil hanya 12 koefisien (elemen) pertamanya saja dan yang lainnya dapat dibuat menjadi 0 (dapat diabaikan). Dengan 12 koefisien *cepstral*, spektrum dari sinyal dapat direkonstruksi dan akan menjadi lebih halus (Roweis, 1998). Dalam grafik pada Gambar 5 terdapat dua buah garis dengan garis yang lebih bergelombang adalah sinyal asli, sedangkan garis yang lebih halus adalah hasil dari analisis *cepstral* dengan 12 koefisien pertama.



Gambar 5. Grafik analisis *cepstral* dibandingkan dengan sinyal asli.

Setelah analisis *cepstral* dilakukan terhadap seluruh *frame* maka akan dihasilkan sebuah matriks yang tiap kolomnya adalah vektor *cepstral* dari tiap *frame* yang hanya diambil 12 koefisien pertamanya saja. Jika K adalah matriks hasil analisis *cepstral* dan v_i adalah vektor *cepstral* dari *frame* ke- i yang berisi 12 koefisien

pertamanya saja dan dengan b adalah banyaknya *frame* maka matriks K dapat diilustrasikan sebagai berikut:

$$K = [v_1 \ v_2 \ v_3 \ \dots \ v_b]$$

Feature Selection

Feature selection bertujuan untuk mengubah dari ruang data ke ruang *feature* yang berdimensi kecil dengan tetap mempertahankan informasi yang penting untuk digunakan dalam aplikasi dan hasil *feature selection* dapat diperbandingkan berdasarkan kemiripan data (Xafopoulos, 2001). Dimensi yang lebih kecil dari data asli akan mempercepat proses-proses selanjutnya.

Salah satu teknik yang dapat digunakan sebagai *feature selection* adalah analisis komponen utama atau *principal component analysis* (PCA). Sesuai dengan tujuan *feature selection*, PCA atau sering disebut dengan transformasi Karhunen-Loeve merupakan cara untuk mereduksi dimensi data tanpa harus kehilangan informasi yang berarti. Hal ini berguna untuk mempersingkat waktu yang diperlukan baik pada saat pembelajaran sistem maupun pada saat digunakan.

Dalam kaitannya dengan langkah-langkah dalam pengenalan pembicara PCA akan menerima input vektor dari hasil *feature extraction*. Hasil *feature extraction* dari sebuah sinyal suara digital adalah sebuah matriks yang tiap kolomnya adalah koefisien-koefisien *cepstral* masing-masing *frame*. Matriks tersebut akan diubah menjadi sebuah vektor. Jika diberikan matriks hasil *feature extraction* C dan v adalah vektor yang merepresentasikan *frame* dengan b adalah banyaknya *frame*, maka matriks C dapat dibentuk menjadi sebuah vektor sebagai berikut:

$$v^T = [v_1 \ v_2 \ \dots \ v_{12}]$$

$$C^T = [v_{11} \ \dots \ v_{112} \ v_{21} \ \dots \ v_{212} \ \dots \ v_{b1} \ \dots \ v_{b12}]$$

PCA melakukan transformasi terhadap C melalui sebuah matriks transformasi P dan menghasilkan matriks hasil transformasi Y atau dalam representasi notasi akan tampak sebagai berikut:

$$Y = PC$$

Jumlah elemen dalam Y dapat disesuaikan sehingga lebih kecil dari C , sehingga dapat dilakukan reduksi dimensi. Y merupakan kombinasi linier dengan vektor-vektor basis dalam matriks P . Hal yang penting dalam PCA

adalah pembentukan matriks transformasi P . Pembentukan P hanya dilakukan satu kali sebelum pembelajaran dilakukan. P akan dibentuk dari sejumlah sinyal suara digital yang telah diproses dengan *feature extraction*. Selain untuk pembentukan P sinyal-sinyal suara digital ini juga akan digunakan untuk pembelajaran. Jika matriks transformasi telah terbentuk maka vektor hasil *feature extraction* dapat langsung dikalikan dengan matriks transformasi sehingga diperoleh vektor baru.

Proses PCA dilakukan sebagai berikut (Shlens, 2003):

Setiap akuisisi satu sinyal suara digital sehingga diperoleh vektor C disebut dengan satu percobaan. Jika terdapat n percobaan dan jika didefinisikan C mempunyai elemen sebanyak m maka dapat dibentuk suatu matriks hasil percobaan-percobaan tersebut. Jika matriks X adalah matriks hasil percobaan-percobaan tersebut maka X dapat ditulis sebagai berikut:

$$X = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mn} \end{bmatrix}$$

c_{ij} : elemen ke- i pada percobaan ke- j

Jika μ_{xi} adalah nilai rata-rata pada baris ke- i maka nilai rata-rata dari tiap baris diberikan sebagai berikut:

$$\mu_{xi} = \frac{1}{n} \sum_{j=1}^n c_{ij}, \quad i = 1, 2, \dots, m$$

Selanjutnya elemen tiap baris dikurangi dengan rata-ratanya sehingga tiap baris dalam X akan mempunyai rata-rata sama dengan nol. Matriks koragam dari X , yaitu S_X dapat dihitung sebagai berikut sesuai dengan persamaan koragam:

$$S_X = \frac{1}{n-1} XX^T$$

Jika diilustrasikan S_X tampak sebagai berikut:

$$S_X = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \dots & \sigma_{1n} \\ \vdots & \sigma_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ \sigma_{n1} & \dots & \dots & \sigma_{nn} \end{bmatrix}$$

σ_{ii} : Nilai ragam baris ke- i dari X

σ_{ij} : Nilai koragam baris ke- i dan baris ke- j dari X

Tampak bahwa S_X adalah sebuah matriks simetri dengan diagonal utamanya adalah ragam dari baris ke- i pada X dan selainnya adalah koragam dari baris ke- i dan ke- j dari X (koragam dari baris ke- i dan baris lainnya). Nilai koragam yang besar merepresentasikan terdapat korelasi yang kuat dan redundansi yang tinggi, sedangkan jika nilai koragam 0 maka tidak ada korelasi. Proses PCA dapat menghilangkan korelasi yang ada sehingga tidak terdapat redundansi. Jika diinginkan suatu hasil transformasi Y yang tidak terdapat redundansi maka hal ini dapat dilakukan melalui analisis pada matriks koragam dari Y , yaitu S_Y . S_Y diberikan oleh persamaan berikut:

$$S_Y = \frac{1}{n-1} YY^T$$

Jika tidak ada redundansi maka pada S_Y bagian selain diagonal utamanya akan bernilai 0 sehingga S_Y akan berupa matriks diagonal.

Asumsi-asumsi yang digunakan dalam PCA adalah (1) seluruh vektor basis dalam P *orthonormal* sehingga P adalah matriks *orthonormal*, dan (2) vektor basis dengan ragam yang terbesar adalah yang paling utama.

Dari hal-hal di atas maka PCA akan mencari suatu matriks *orthonormal* P dengan $Y = PX$ sehingga S_Y diagonal. Baris-baris dari P adalah komponen utama dari X . Analisis S_Y diberikan sebagai berikut:

$$\begin{aligned} S_Y &= \frac{1}{n-1} YY^T \\ &= \frac{1}{n-1} (PX)(PX)^T \\ &= \frac{1}{n-1} PXX^T P^T \\ &= \frac{1}{n-1} P(XX^T)P^T \\ S_Y &= \frac{1}{n-1} PAP^T \end{aligned}$$

Di atas didefinisikan suatu matriks baru $A \equiv XX^T$ dan A adalah matriks simetri yang dibuktikan oleh suatu teorema bahwa jika W adalah sembarang matriks maka WW^T dan $W^T W$ keduanya adalah simetri (Shlens, 2003). Teorema-teorema lain yang digunakan adalah (Shlens, 2003):

1. Sebuah matriks dikatakan simetri jika dan hanya jika dapat didiagonalkan secara

orthogonal. Hal ini berarti jika A dapat didiagonalkan secara *orthogonal* maka A adalah matriks simetri. Hipotesis yang digunakan dalam pendidagonalan secara *orthogonal* adalah terdapat suatu matriks E sehingga $A = EDE^T$, dengan D adalah matriks diagonal dan E adalah sebuah matriks tertentu yang mendidagonalkan A . Jika dihitung A^T sebagai berikut:

$$A^T = (EDE^T)^T = E^{TT}D^TE^T = EDE^T = A$$

maka jika A dapat didiagonalkan secara *orthogonal*, A adalah matriks simetri.

2. Sebuah matriks simetri dapat didiagonalkan oleh matriks vektor ciri-vektor cirinya yang *orthogonal*. Misalnya A adalah sebuah matriks simetri dan $E = [e_1 \ e_2 \ \dots \ e_n]$ dengan tiap kolom dari E adalah vektor ciri-vektor ciri dari A . Teorema ini perluasan dari teorema 1 dan teorema ini mengemukakan bahwa terdapat matriks diagonal D sehingga $A = EDE^T$. Misalnya diberikan D suatu matriks diagonal yang memuat akar ciri-akar ciri dari A dengan akar ciri ke- i diletakkan pada posisi ke- ii maka menurut persamaan akar ciri dapat didefinisikan $AE = ED$ atau $A = EDE^{-1}$. Karena A simetri maka vektor ciri-vektor cirinya akan *orthogonal* dan karena itu maka E *orthogonal*. Jika E *orthogonal* maka $E^{-1} = E^T$ sehingga diperoleh $A = EDE^T$.

Karena A adalah matriks simetri, dari teorema-teorema di atas dapat didefinisikan $A = EDE^T$ dengan E adalah matriks yang kolomnya adalah vektor ciri-vektor ciri dari A dan D adalah matriks diagonal dengan elemen ke- ii pada D adalah akar ciri-akar ciri yang bersesuaian dengan vektor ciri-vektor ciri pada E . Jika dipilih P dengan tiap baris dari P adalah vektor ciri dari XX^T maka $P \equiv E^T$ sehingga diperoleh $A = P^TDP$ dan $P^{-1} \equiv P^T$ karena P *orthogonal*. S_Y dapat ditulis ulang sebagai berikut:

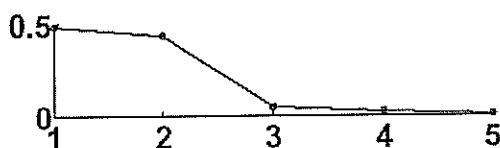
$$\begin{aligned} S_Y &= \frac{1}{n-1} PAP^T \\ &= \frac{1}{n-1} P(P^TDP)P^T \\ &= \frac{1}{n-1} (PP^T)D(PP^T) \\ &= \frac{1}{n-1} (PP^{-1})D(PP^{-1}) \end{aligned}$$

$$= \frac{1}{n-1} D$$

Dapat dilihat bahwa pemilihan P dapat mendiagonalkan S_Y dan akan diperoleh hasil transformasi yang tidak saling berkorelasi.

Dalam praktiknya untuk melakukan PCA dari suatu set percobaan yang diberikan dalam matriks X hanya perlu dilakukan 2 hal, yaitu (1) menghitung rata-rata tiap baris kemudian mengurangi elemen tiap baris dengan rata-ratanya sehingga diperoleh X baru, dan (2) menghitung vektor ciri-vektor ciri dan akar ciri-akar ciri dari XX^T . Vektor ciri-vektor ciri dari XX^T akan membentuk P dan besarnya ragam dari P dapat diperoleh dari akar ciri-akar ciri XX^T .

Reduksi dimensi dapat dilakukan dengan tidak mengambil seluruh komponen utama yang ada dengan melihat besarnya ragam pada S_Y . Semakin besar ragam maka semakin berpengaruh komponen utama tersebut. Jika dimisalkan terdapat n komponen utama, maka kita dapat mengambil sebanyak q komponen utama ($q \leq n$) sehingga dapat dibentuk suatu matriks berordo $q \times n$ yang terdiri dari sejumlah q komponen utama terbesar saja. Menurut Shlens (2003), pemilihan besarnya q dapat dilakukan dengan melihat grafik akar ciri. Sebagai contoh diberikan grafik akar ciri seperti pada Gambar 6, dapat dipilih banyaknya komponen utama yang sebaiknya digunakan adalah dua buah karena setelah itu nilai kontribusinya terhadap data menurun dengan tajam (berkontribusi kecil).



Gambar 6. Contoh grafik akar ciri yang telah diurutkan.

Pembentukan Model Referensi Pembicara dan Pencocokan Pola

Dalam pengenalan pembicara pembentukan model referensi pembicara dan pencocokan pola adalah dua tahap yang sangat berkaitan. Pembentukan model referensi pembicara akan membentuk suatu model referensi yang akan digunakan untuk pencocokan pola. Salah satu teknik yang dapat digunakan dalam pencocokan pola adalah JST. JST akan melakukan pembelajaran untuk membentuk suatu model referensi, kemudian JST yang telah melakukan

pembelajaran tersebut dapat digunakan untuk pencocokan pola.

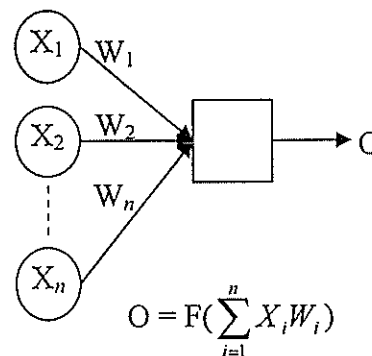
Sebuah jaringan syaraf tiruan adalah sebuah sistem pemrosesan informasi yang mempunyai karakteristik serupa dengan jaringan syaraf biologi (Fausett, 1994). Sebuah JST direpresentasikan oleh sebuah set *node-node* dan panah-panah penghubung. Sebuah *node* mewakili sebuah neuron dan sebuah panah mewakili hubungan antarneuron dengan arah panah menunjukkan aliran sinyal.

Setiap *node* menerima sebuah set *input* yang akan dikalikan dengan *weight* (bobot) yang dianalogikan sebagai kuat lemahnya *synapsis* dalam sel biologi. Jumlah total dari seluruh *input* yang telah dikalikan bobot akan menentukan level pengaktifan *node* tersebut. Dalam representasi notasi setiap *input* X_i dikalikan bobot W_i sehingga total *input*-nya akan seperti ekspresi berikut:

$$\sum_i X_i W_i$$

Total *input* tersebut kemudian diproses oleh suatu fungsi pengaktifan dan akan menghasilkan suatu *output*. Contoh sebuah JST dapat dilihat pada Gambar 7.

Kecerdasan dari JST diperoleh dari perilaku kolektif neuron-neuronnya yang masing-masing melakukan operasi yang sangat terbatas. JST dapat menemukan solusi dengan cepat karena tiap neuron dapat bekerja secara paralel.



Gambar 7. Model JST.

Salah satu model JST yang dapat digunakan untuk pencocokan pola adalah JST *backpropagation*. JST *backpropagation* dikembangkan oleh Rumelhart, Hinton dan Williams yang dipopulerkan dalam buku *Parallel Distributed Processing*. JST *backpropagation* menggunakan arsitektur *multi-layer perceptron*

dan pembelajaran *backpropagation*. Walaupun JST *backpropagation* membutuhkan waktu yang relatif lama untuk pembelajaran tetapi bila pembelajaran telah selesai dilakukan, JST akan dapat mengenali suatu pola dengan cepat.

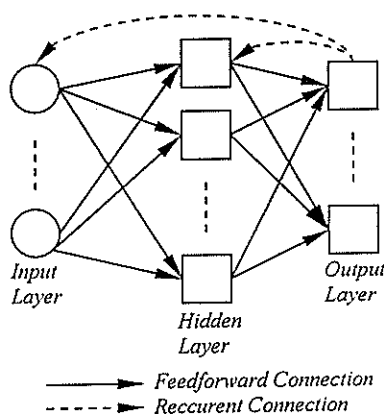
Beberapa karakteristik dari JST *backpropagation* adalah sebagai berikut:

- Jaringan *multi-layer*. JST *backpropagation* mempunyai lapisan *input*, lapisan tersembunyi dan lapisan *output* (Gambar 8) dan setiap neuron pada satu lapisan menerima *input* dari semua neuron pada lapisan sebelumnya.
- Fungsi pengaktifan. Fungsi pengaktifan akan menghitung *input* yang diterima oleh suatu neuron, kemudian neuron tersebut meneruskan hasil dari fungsi pengaktifan ke neuron berikutnya, sehingga fungsi pengaktifan berfungsi sebagai penentu kuat lemahnya sinyal yang dikeluarkan oleh suatu neuron. Beberapa fungsi pengaktifan yang sering digunakan dalam JST *backpropagation* adalah:
 - Fungsi sigmoid biner. Fungsi sigmoid biner memiliki rentang 0–1 dengan fungsi sebagai berikut:

$$f(x) = \frac{1}{1 + \exp(-x)}$$

- Fungsi sigmoid bipolar. Fungsi sigmoid bipolar memiliki rentang -1-1 dengan fungsi sebagai berikut:

$$f(x) = \frac{2}{1 + \exp(-x)} - 1$$



Gambar 8. Model JST *backpropagation*.

Algoritme pembelajaran JST *backpropagation* bersifat iteratif dan didesain untuk meminimalkan

mean square error (MSE) antara *output* yang dihasilkan dengan *output* yang diinginkan. Langkah-langkah algoritme pembelajaran JST *backpropagation* yang diformulasikan oleh Rumelhart, Hinton, dan Williams secara singkat adalah sebagai berikut (selengkapnya dapat dilihat pada Lampiran 1):

- Inisialisasi bobot. Inisialisasi dapat dilakukan secara acak atau melalui metode Nguyen-Widrow.
- Perhitungan nilai pengaktifan. Tiap neuron menghitung nilai pengaktifan dari *input* yang diterimanya. Pada lapisan *input* nilai pengaktifan adalah fungsi identitas. Pada lapisan tersembunyi dan *output* nilai pengaktifan dihitung melalui fungsi pengaktifan.
- Penyesuaian bobot. Penyesuaian bobot dipengaruhi oleh besarnya nilai kesalahan (*error*) antara target *output* dan nilai *output* jaringan saat ini.
- Iterasi akan terus dilakukan sampai kriteria *error* tertentu dipenuhi.

Untuk mengimplementasikan algoritme di atas (pembelajaran), JST harus memiliki suatu set data pembelajaran. Data pembelajaran harus mencakup seluruh jenis pola yang ingin dikenali agar JST nantinya dapat mengenali seluruh pola-pola yang ada. Dalam kaitannya dengan sistem pengenalan pembicara, data pembelajaran harus mencakup seluruh pembicara yang ada. Dalam JST, semakin banyak contoh suatu pola dalam pembelajaran maka JST akan semakin baik mengenali pola tersebut. Untuk itu akan lebih baik jika tiap pembicara mengucapkan lebih dari satu kali pengulangan untuk nantinya digunakan dalam pembelajaran JST.

JST akan menerima data *input* berupa vektor. Jika dimensi vektor terlalu besar maka JST akan bekerja lebih lambat. Dalam pengenalan pembicara setiap sinyal suara digital akan diproses terlebih dahulu dengan teknik-teknik *feature extraction* dan *feature selection* sehingga dimensi data akan tereduksi.

Dalam pembelajaran seluruh set data pembelajaran akan diproses sehingga JST akan membentuk suatu model referensi bagi seluruh pola-pola yang ada.

Dalam representasi numerik data *input* untuk pembelajaran JST akan berupa sebuah matriks dan jika dimisalkan setiap kolom merepresentasikan sebuah data berupa vektor

maka matriks tersebut misalnya J dengan n data akan tampak sebagai berikut:

$$J = [\text{data}_1, \text{data}_2, \text{data}_3, \dots, \text{data}_n]$$

Setiap data dalam J mempunyai pasangan target yang akan digunakan dalam pembelajaran. Setiap kolom dalam J di-input-kan ke dalam JST satu per satu.

Hal pertama yang dilakukan pada tahap pembelajaran adalah inisialisasi bobot. Inisialisasi bobot dapat dilakukan secara acak atau melalui metode Nguyen-Widrow (selengkapnya dapat dilihat pada algoritme *backpropagation* dalam Lampiran 1).

Pada lapisan *input* setiap elemen vektor *input* akan diterima oleh sebuah neuron sehingga jumlah neuron pada lapisan ini akan sama dengan banyaknya elemen vektor *input*. Lapisan *input* memiliki fungsi pengaktifan berupa fungsi identitas sehingga hanya berfungsi meneruskan *input* yang diterima ke lapisan berikutnya.

Lapisan tersembunyi akan menerima *output* yang dikeluarkan oleh lapisan *input*. Setiap neuron pada lapisan tersembunyi menerima *input* dari seluruh neuron pada lapisan *input* dikalikan dengan bobotnya. *Input* yang masuk dihitung dengan suatu fungsi pengaktifan dan hasil dari fungsi pengaktifan ini akan menjadi *output* tiap neuron pada lapisan tersembunyi. Banyaknya neuron pada lapisan tersembunyi dapat bervariasi dan dapat dianggap cukup jika JST dapat mengenali pola-pola yang ada dengan cukup baik (Fu, 1994).

Lapisan *output* akan menerima *output* yang dikeluarkan oleh lapisan tersembunyi. Setiap neuron pada lapisan *output* menerima *input* dari seluruh neuron pada lapisan tersembunyi dikalikan dengan bobotnya. *Input* yang masuk juga dihitung dengan suatu fungsi pengaktifan dan hasil dari fungsi pengaktifan ini akan menjadi *output* tiap neuron. Untuk kemudahan dan hasil yang lebih baik jumlah neuron pada lapisan *output* dapat ditentukan sama dengan jumlah pola yang ada. Fase dari data pertama kali masuk dalam lapisan *input* sampai lapisan *output* memberikan hasil merupakan fase *feedforward*.

Jika telah diperoleh hasil pada lapisan *output* maka hasil ini akan dibandingkan dengan target pasangan untuk data yang masuk. Dari perbedaan nilai antara target yang diinginkan dengan hasil saat ini dapat dihitung suatu nilai kesalahan. Untuk seluruh data yang ada dapat dihitung suatu nilai total kesalahan. Fase ini merupakan fase kalkulasi *error*.

Nilai *error* yang diperoleh akan digunakan untuk memperbaiki nilai bobot-bobot pada JST sehingga JST akan semakin baik mengenali pola-pola yang ada. Fase ini disebut dengan fase penyesuaian bobot.

Setelah bobot-bobot diperbaiki data pembelajaran kembali di-input-kan dalam jaringan dan kembali diperoleh nilai *error* dan bobot akan kembali diperbaiki sehingga pembelajaran akan bersifat iteratif. Iterasi dapat dihentikan jika kriteria *error* tertentu dipenuhi atau jumlah *epoch* (satu *cycle* seluruh data pembelajaran melewati jaringan) tertentu dipenuhi. Kriteria henti dengan menggunakan suatu nilai *error* tertentu mengimplikasikan jika nilai *error* cukup kecil maka jaringan akan cukup baik untuk mengenali pola-pola yang ada. Namun nilai *error* yang terlalu kecil akan membuat jaringan terlalu spesifik mengenali pola-pola pembelajaran (*overtrained*) dan kemampuannya mengenali pola-pola baru yang serupa tetapi tidak identik dengan pola pembelajaran akan menurun (Fu, 1994).

Setelah pembelajaran selesai akan dihasilkan suatu JST yang telah dapat mengenali pola-pola yang ada dan siap digunakan untuk pencocokan pola. JST ini akan menerima *input* berupa vektor yang masuk ke dalam lapisan *input*-nya kemudian setelah diolah maka akan diperoleh hasil pada lapisan *output*. Jika jumlah neuron *output* ditentukan sama dengan banyaknya jenis pola yang ada maka tiap neuron merepresentasikan tiap pola.

JST *backpropagation* dikenal sebagai JST yang dapat memberikan respon yang cukup baik untuk pola-pola yang serupa tetapi tidak identik dengan pola pembelajaran (Fausett, 1994). Pengujian JST untuk pengenalan pola dapat dilakukan dengan generalisasi, yaitu jumlah (dalam %) pola yang berhasil diklasifikasi dengan benar oleh JST. Generalisasi diberikan oleh persamaan berikut (Herryadie, 1999):

$$\text{Generalisasi} = \frac{\text{Jumlah pola yang dikenali}}{\text{Jumlah seluruh pola}} \times 100\%$$

Pembuatan Keputusan

Sistem pengenalan pembicara mempunyai dua buah model pembuatan keputusan, yaitu untuk sistem identifikasi dan untuk sistem verifikasi. Proses pembuatan keputusan terkait erat dengan teknik pencocokan pola yang digunakan. Pembuatan keputusan identifikasi dapat

dianalogikan sebagai masalah klasifikasi pola dengan tiap kelas merepresentasikan tiap pembicara. Pada masalah pengenalan pola, JST akan memberikan skor bagi pola yang masuk untuk semua kelas yang ada. Metode nilai maksimum melakukan pembuatan keputusan dengan melihat kelas yang mempunyai skor maksimum (Riadi, 2001), dan pola yang masuk akan diklasifikasi ke dalam kelas (pembicara) tersebut.

Pembuatan keputusan untuk verifikasi pada pengenalan pembicara akan menentukan diterima atau tidaknya data suara yang masuk. Salah satu metode yang dapat digunakan untuk melakukan verifikasi adalah metode *threshold* (Ho, 1998). Pembuatan keputusan dilakukan melalui perbandingan skor hasil pencocokan pola dengan besaran *threshold*. Jika skor lebih besar atau sama dengan *threshold* maka verifikasi diterima dan jika lebih kecil maka verifikasi ditolak. *Threshold* yang digunakan dapat berlaku untuk seluruh pembicara atau dapat juga tiap pembicara memiliki *threshold* yang berbeda-beda. *Threshold* yang berbeda-beda untuk tiap pembicara menawarkan fleksibilitas karena besarnya nilai *threshold* dapat diatur sesuai kebutuhan dan perubahan *threshold* pada satu pembicara tidak akan mempengaruhi *threshold* pembicara lainnya. Salah satu metode penentuan *threshold* yang diajukan oleh Ho (1998) untuk sistem verifikasi dan telah disesuaikan untuk JST *backpropagation* adalah sebagai berikut:

- Bentuk sebuah matriks **O** menggunakan sistem identifikasi yang tiap kolomnya berisi hasil *output* JST untuk tiap data pengujian. Data yang tidak berhasil diidentifikasi dengan benar tidak dimasukkan ke dalam **O**. Karena data dalam matriks **O** dapat diidentifikasi dengan benar, nilai maksimum dari tiap kolom merepresentasikan identitas pembicara.
- Untuk tiap set *sample* pembicara bentuk sebuah vektor **m** yang berisi nilai maksimum dari tiap kolom pada matriks **O**. Misal dalam **O** terdapat 5 *sample* pembicara 1, maka vektor **m** untuk pembicara 1 akan mempunyai elemen sebanyak 5 elemen. Vektor **m** akan berjumlah sama dengan jumlah pembicara.
- Setelah itu elemen-elemen vektor **m** diurutkan. Untuk tiap pembicara dipilih satu nilai dari vektor **m** yang bersesuaian sebagai *threshold* bagi pembicara tersebut.

Pengujian Pengenalan Pembicara

Pengujian sistem pengenalan pembicara terbagi menjadi dua bagian, yaitu pengujian sistem identifikasi dan pengujian sistem verifikasi. Pengujian sistem identifikasi akan melibatkan pengujian pembelajaran JST. Beberapa hal yang dilihat pada pengujian identifikasi adalah sebagai berikut (Fu, 1994):

- Kinerja sistem. Kinerja sistem diukur dari generalisasi yang dihasilkan sistem terhadap pola-pola pengujian.
- Efisiensi. Efisiensi dapat diukur dari waktu yang diperlukan sistem untuk melakukan pengenalan.

Untuk sistem verifikasi pengujian dilakukan dengan melihat nilai *false acceptance rate* (FAR), *false rejection rate* (FRR), dan *equal error rate* (EER) serta pembentukan grafik *detection error tradeoff* (DET) (Xafopoulos, 2001). FAR adalah besarnya peluang sistem untuk melakukan kesalahan penerimaan (menerima *impostor*). FRR adalah besarnya peluang sistem untuk melakukan kesalahan penolakan (menolak orang yang benar). Nilai dari FAR dan FRR diberikan oleh persamaan berikut:

$$FAR = \frac{\# \text{ false acceptances}}{\# \text{ false claims}} \times 100\%$$

$$FRR = \frac{\# \text{ false rejections}}{\# \text{ true claims}} \times 100\%$$

Dari nilai FAR dan FRR yang berurutan dapat dihitung nilai EER, yaitu suatu nilai sedemikian sehingga $FAR \approx FRR$. Nilai dari EER diberikan oleh persamaan berikut:

$$EER = \frac{FRR_j \cdot FAR_{j+1} - FRR_{j+1} \cdot FAR_j}{(FAR_{j+1} - FAR_j) - (FRR_{j+1} - FRR_j)}$$

$$FAR_{i+1} \geq FAR_i \wedge FRR_{i+1} \leq FRR_i, \forall i$$

$$FAR_j \leq FRR_{j+1} \wedge FAR_{j+1} \geq FRR_j$$

dengan FAR_i dan FRR_i adalah FAR dan FRR dengan konfigurasi *threshold* ke-*i*, dan FAR_j dan FRR_j adalah FAR dan FRR saat ini. Setiap perubahan konfigurasi nilai *threshold* akan mempengaruhi nilai FAR dan FRR dengan FAR semakin besar maka FRR semakin kecil dan sebaliknya. Dari perubahan-perubahan nilai *threshold* dapat dibentuk grafik DET, yaitu suatu grafik yang menggambarkan *error tradeoff* antara FAR dan FRR.

METODE PENELITIAN

Langkah-langkah Pengembangan Sistem

Langkah-langkah untuk mengembangkan sistem pengenalan pembicara terdiri atas (1) akuisisi data suara digital, (2) *feature extraction* dan *feature selection*, (3) pembentukan model referensi pembicara dan pencocokan pola, dan (4) pembuatan keputusan.

Akuisisi Data Suara Digital. Akuisisi data dilakukan menggunakan mikrofon. Pengguna sistem akan mengucapkan kata yang telah ditentukan sebelumnya. Data audio yang diperoleh akan diubah menjadi bentuk digital (vektor) menggunakan proses *sampling* dengan perangkat lunak MATLAB 6.5. Perecaman suara dilakukan selama 3 detik dengan frekuensi *sampling (fs)* 11.025 Hz (dalam 1 detik diperoleh data sebanyak 11.025 data).

Akuisisi data dilakukan pada beberapa tahap. Pada tahap pertama dilakukan akuisisi data untuk pembelajaran sistem. Pada tahap kedua akuisisi data dilakukan untuk menguji sistem identifikasi. Data pada tahap kedua sebagian juga akan digunakan untuk penentuan *threshold* dan pengujian verifikasi. Pada tahap ketiga dilakukan akuisisi data yang akan digunakan sebagai *impostor* pada sistem verifikasi.

Feature extraction dan feature selection. Sebelum dilakukan *feature extraction* akan dilakukan tahap-tahap sebagai berikut:

1. *Frame blocking.* Dalam penelitian ini digunakan *frame* dengan panjang *frame (n)* sebesar 256 *sample* atau setara dengan 23,22 milidetik dan besaran $m = 100$. Tiap *frame* akan saling *overlap* sebanyak $n - m$ *sample*.
2. *Frame windowing* menggunakan *hamming window* dengan panjang *window* sama dengan panjang *frame* yaitu 256 *sample*.

Setelah data telah terbagi dalam *frame-frame* dan telah dikalikan dengan *hamming window*, dilakukan *feature extraction* menggunakan analisis *cepstral*. Dari analisis *cepstral* akan diperoleh 12 koefisien untuk tiap *frame*. Hasil dari *feature extraction* berupa matriks dan tiap kolom merupakan representasi dari tiap *frame*.

Selanjutnya adalah *feature selection* menggunakan PCA. Matriks yang diperoleh dari *feature extraction* diubah menjadi bentuk vektor. Pembentukan matriks transformasi akan menggunakan seluruh data hasil akuisisi pada tahap pertama yang telah diproses dengan *feature extraction*. Data ini berbentuk sebuah matriks dan

dicari matriks koragamnya. Matriks transformasi adalah vektor ciri-vektor ciri matriks koragam tersebut dan dengan grafik akar cirinya maka dapat dipilih jumlah komponen utama yang akan digunakan. Setelah matriks transformasi terbentuk maka data yang masuk diproses menggunakan matriks transformasi untuk kemudian diteruskan ke JST.

Pembentukan Model Referensi Pembicara dan Pencocokan Pola. Pembentukan model referensi pembicara dan pencocokan pola dilakukan menggunakan JST *backpropagation*. Arsitektur yang digunakan untuk JST *backpropagation* adalah *multi-layer perceptron* dengan satu lapisan tersembunyi. JST terlebih dahulu dilatih untuk membentuk model referensi pembicara. Setelah tahap pembelajaran selesai dilakukan, JST dapat digunakan untuk melakukan pencocokan pola.

Jumlah neuron pada lapisan *output* sama dengan jumlah kelas yang akan diklasifikasi (jumlah pembicara), sedangkan jumlah neuron pada lapisan tersembunyi jumlahnya dapat bervariasi dan neuron pada lapisan *input* jumlahnya bergantung pada hasil PCA.

Untuk inisialisasi bobot digunakan inisialisasi Nguyen-Widrow dan fungsi pengaktifan dengan sigmoid biner. Penggunaan sigmoid biner dianggap sesuai untuk pengenalan dengan selangnya yang berada antara 0 sampai 1. Secara matematis sigmoid biner juga lebih cepat dibandingkan dengan sigmoid bipolar karena operasi yang dilakukan lebih sedikit. Target menggunakan nilai 1 pada neuron *output* untuk pembicara yang bersesuaian dan 0 untuk yang lainnya. Toleransi galat akan ditentukan pada 0,01; 0,001 dan 0,0001 dan laju pembelajaran yang digunakan adalah 0,1; 0,2 dan 0,3. Dalam penelitian ini akan dilihat kombinasi toleransi galat dan laju pembelajaran yang optimal. Jumlah *epoch* maksimal ditetapkan sebanyak 5.000. Hal ini perlu dilakukan sebagai kriteria henti jaringan di samping toleransi galat untuk membatasi waktu yang disediakan bagi jaringan dalam melakukan pembelajaran.

Pengujian untuk menentukan jumlah neuron tersembunyi dilakukan dengan laju pembelajaran 0,1 dan toleransi galat 0,001. Jumlah awal neuron tersembunyi dibuat sama dengan 10. Toleransi galat yang cukup kecil diharapkan akan memberikan hasil yang cukup baik. Jika ternyata JST gagal mencapai kekonvergenan maka akan dilakukan penambahan jumlah neuron tersembunyi sampai kekonvergenan dicapai

sedangkan jika JST berhasil mencapai kekonvergenan maka akan dilihat generalisasinya dan dilakukan penambahan neuron tersembunyi. Jika ternyata generalisasi yang dihasilkan tidak jauh berbeda dengan jumlah neuron tersembunyi sebelumnya maka JST telah sampai pada batas optimal. Penambahan kembali neuron tersembunyi tidak akan menambah generalisasi dan hanya akan menambah *cost* untuk melakukan perhitungan.

Untuk kemudahan dan fleksibilitas digunakan MATLAB 6.5 untuk membangun JST dengan fungsi pembelajaran *traingdx* yang terdapat dalam *neural network toolbox*. Fungsi *traingdx* menyediakan berbagai fasilitas dalam melakukan pembelajaran JST seperti penentuan besaran-besaran peubah yang akan diambil dan beberapa peubah tambahan yang merupakan modifikasi dari pembelajaran *backpropagation*, tetapi dalam penelitian ini yang digunakan adalah tetap pembelajaran *backpropagation* standar.

Pembuatan Keputusan. Pada proses identifikasi, pembuatan keputusan dilakukan dengan metode nilai maksimum. Jika neuron *output* ke-*n* merupakan neuron dengan nilai maksimum maka data yang masuk dikenali sebagai pembicara ke-*n*. Contoh jika neuron pertama pada lapisan *output* bernilai 1 dan yang lainnya 0 maka *input* diidentifikasi sebagai pembicara pertama.

Pada proses verifikasi, pembuatan keputusan dilakukan dengan metode *threshold*. Jika seseorang mengklaim bahwa dirinya adalah pembicara 1 dan *output* pada neuron pertama lebih besar atau sama dengan *threshold* maka dikatakan benar bahwa orang tersebut adalah pembicara 1 dan jika ternyata *output* pada neuron pertama lebih kecil dari *threshold* maka dikatakan bahwa orang tersebut bukan pembicara 1. *Threshold* yang digunakan berbeda-beda untuk tiap pembicara. Dengan metode satu pembicara satu *threshold* maka untuk tiap pembicara besarnya *threshold* dapat disesuaikan dengan skor dari hasil pengujian identifikasi.

Data Teknis

Berikut ini disajikan data teknis yang digunakan dalam penelitian dengan Tabel 1 untuk data proses sinyal digital, Tabel 2 untuk data JST, dan Tabel 3 contoh definisi target JST untuk 13 pembicara.

Tabel 1. Data proses sinyal digital

Karakteristik	Spesifikasi
Frekuensi <i>sampling</i>	11.025 Hz
Durasi perekaman	3 detik
Panjang <i>frame</i> (<i>n</i>)	256 <i>sample</i>
<i>Overlap</i> (<i>n - m</i>)	(256 - 100) <i>sample</i>
<i>Frame windowing</i>	<i>Hamming window</i>
Koefisien <i>cepstral</i>	12 koefisien

Tabel 2. Data JST

Karakteristik	Spesifikasi
Arsitektur	1 lapisan tersembunyi
Neuron <i>input</i>	Hasil <i>feature extraction & feature selection</i>
Neuron tersembunyi	Dari pengujian (dimulai dari 10)
Neuron <i>output</i>	Jumlah pembicara
Inisialisasi bobot	Nguyen-Widrow
Fungsi pengaktifan	Sigmoid biner
Toleransi galat	0,01; 0,001 dan 0,0001
Laju pembelajaran	0,1; 0,2 dan 0,3
<i>Epoch</i> maksimum	5.000 <i>epoch</i>
<i>Sample</i> pembelajaran tiap pembicara	5 <i>sample</i>
<i>Sample</i> pengujian tiap pembicara	5 <i>sample</i>

Tabel 3. Contoh definisi target JST untuk 13 pembicara

No	Target	Representasi suara
1	1 0 0 0 0 0 0 0 0 0 0 0 0	Pembicara ke-1
2	0 1 0 0 0 0 0 0 0 0 0 0 0	Pembicara ke-2
3	0 0 1 0 0 0 0 0 0 0 0 0 0	Pembicara ke-3
4	0 0 0 1 0 0 0 0 0 0 0 0 0	Pembicara ke-4
5	0 0 0 0 1 0 0 0 0 0 0 0 0	Pembicara ke-5
6	0 0 0 0 0 1 0 0 0 0 0 0 0	Pembicara ke-6
7	0 0 0 0 0 0 1 0 0 0 0 0 0	Pembicara ke-7
8	0 0 0 0 0 0 0 1 0 0 0 0 0	Pembicara ke-8
9	0 0 0 0 0 0 0 0 1 0 0 0 0	Pembicara ke-9
10	0 0 0 0 0 0 0 0 0 1 0 0 0	Pembicara ke-10
11	0 0 0 0 0 0 0 0 0 0 1 0 0	Pembicara ke-11
12	0 0 0 0 0 0 0 0 0 0 0 1 0	Pembicara ke-12
13	0 0 0 0 0 0 0 0 0 0 0 0 1	Pembicara ke-13

Spesifikasi Perangkat Keras dan Lunak

Sistem pengenalan pembicara ini dibangun dan diuji coba menggunakan sistem dengan spesifikasi sebagai berikut:

1. Perangkat keras
 - Prosesor AMD Athlon XP 1,8 GHz
 - RAM 256 MB
 - Kartu suara *onboard* NVIDIA nForce MCP
 - Mikrofon jenis *desktop*
2. Perangkat lunak
 - Sistem operasi Microsoft Windows XP
 - MATLAB versi 6.5 dilengkapi dengan *toolbox-toolbox* sebagai berikut:
 - *Signal processing toolbox*
 - *Neural network toolbox*

Pengujian Sistem

Untuk sistem identifikasi, peubah-peubah yang akan dilihat antara lain adalah waktu dan jumlah *epoch* yang dibutuhkan jaringan untuk mencapai kekonvergenan (efisiensi) dan waktu untuk melakukan identifikasi untuk seluruh data pengujian beserta generalisasi yang dihasilkan JST (kinerja sistem).

Evaluasi kinerja sistem verifikasi akan dilihat dari nilai FAR, FRR, dan EER yang dihasilkan serta grafik DET yang terbentuk.

HASIL DAN PEMBAHASAN

Akuisisi Data Suara Digital

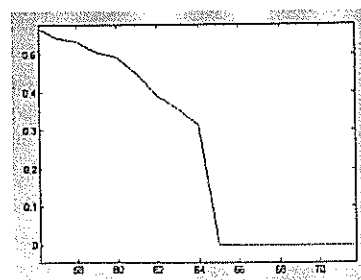
Data suara yang diperoleh berjumlah 208 data yang terdiri dari 130 data dari tahap pertama dan kedua dan 78 sisanya dari tahap ketiga. Data tahap pertama digunakan untuk pembelajaran, data tahap kedua digunakan untuk pengujian identifikasi dan sebagian juga digunakan untuk penentuan *threshold* dan mengevaluasi kinerja verifikasi, sedangkan data tahap ketiga adalah data yang digunakan sebagai *impostor* pada verifikasi. Data sebesar 130 data dari tahap pertama dan kedua merupakan data suara yang diperoleh dari 13 orang dengan 10 data suara tiap orangnya. Dari 10 data tersebut 5 data digunakan untuk pembelajaran sedangkan sisanya digunakan untuk pengujian identifikasi dan verifikasi. Dari hasil kombinasi *fs* yang digunakan dan lamanya durasi perekaman maka tiap data suara berupa vektor dengan 33.075 elemen (11.025×3). Jumlah pembicara yang diambil pada tahap

pertama dan kedua menentukan besar jumlah neuron *output* pada JST. Dengan diperolehnya data dari 13 orang maka jumlah neuron *output* pada JST berjumlah 13 neuron. Data tahap ketiga yang berjumlah 78 data suara diperoleh dari dua orang *impostor* dengan masing-masing mengucapkan tiga pengulangan untuk tiap pembicara.

Feature Extraction dan Feature Selection

Proses *feature extraction* menerima *input* sinyal suara digital dengan tiap sinyal berupa vektor kolom yang berdimensi 33.075 dari hasil perekaman selama 3 detik dengan *fs* sama dengan 11.025. Pada tahap pembelajaran diolah sebanyak 65 data suara hasil dari perekaman 13 orang dengan 5 *sample* setiap orangnya. Setelah melalui tahap-tahap *feature extraction*, tiap data suara menghasilkan vektor *feature* yang dalam penelitian ini berdimensi 3.912. Untuk seluruh 65 data pelatihan waktu yang dibutuhkan untuk melakukan *feature extraction* adalah 6,515 detik.

Proses *feature selection* dilakukan terhadap matriks yang tiap kolomnya adalah vektor *feature* tiap data suara. Proses *feature selection* dengan PCA melibatkan pembentukan matriks koragam serta pembentukan akar ciri dan vektor ciri. Setelah dilakukan, proses PCA membutuhkan waktu selama $1,2537 \times 10^3$ detik atau 20,895 menit. Waktu yang dibutuhkan cukup lama karena besarnya dimensi matriks *input* yaitu 3.912×65 . Proses PCA menghasilkan matriks baru hasil transformasi, nilai akar ciri dari matriks koragam beserta vektor cirinya. Setelah dilihat pada grafik akar ciri ternyata setelah akar ciri yang ke-64 terjadi penurunan kontribusi yang signifikan seperti terlihat pada Gambar 9. Komponen utama yang diambil (matriks transformasi) diperoleh dari 64 vektor ciri yang bersesuaian dengan 64 akar ciri pertama. Dimensi baru hasil proses PCA ini nantinya akan berimplikasi pada jumlah neuron *input* JST. Jumlah neuron *input* JST bergantung pada dimensi *output* PCA.



Gambar 9. Grafik akar ciri data pembelajaran.

Pembentukan Model Referensi Pembicara dan Pencocokan Pola serta Hasil Identifikasi (Pembelajaran dan Generalisasi JST)

Pada tahap pembelajaran JST menyesuaikan besar bobot-bobotnya untuk mencapai kekonvergenan untuk membentuk suatu model referensi bagi pola-pola lain. Semakin besar jumlah *input* yang diterima maka semakin besar pula waktu pembelajaran yang dibutuhkan JST karena bobot yang harus disesuaikan nilainya juga semakin banyak. Dengan dimensi yang tereduksi pada proses sebelumnya diharapkan waktu yang dibutuhkan JST untuk melakukan pembelajaran juga akan semakin singkat. Dalam penelitian ini akan dilihat perilaku JST dalam mencapai kekonvergenan dengan mengubah-ubah peubah yang berpengaruh terhadap kekonvergenan, yaitu laju pembelajaran dan toleransi galat. Pengaruh banyaknya neuron tersembunyi terhadap konvergensi dan generalisasi juga akan dilihat. Tabel hasil pengujian jumlah neuron tersembunyi dapat dilihat pada Lampiran 2.

Pengujian selanjutnya yang dilakukan merupakan kombinasi dari nilai-nilai laju pembelajaran dan toleransi galat. Total terdapat 9 percobaan kombinasi dari 3 nilai toleransi galat yang dicobakan dan 3 nilai laju pembelajaran. Hal-hal yang dilihat adalah jumlah *epoch* untuk mencapai kekonvergenan, waktu yang dibutuhkan untuk mencapai jumlah *epoch* tersebut, dan nilai generalisasi yang didapat dari pembelajaran yang dilakukan beserta waktunya. Pada percobaan pertama dengan toleransi galat 0,01 dengan laju pembelajaran 0,1 jaringan konvergen dalam 74 *epoch* dengan waktu 1,985 detik dan generalisasi yang dihasilkan sebesar 86,1538% (56 dari 65 data pengujian berhasil diidentifikasi dengan benar). Pada Tabel 4 ditunjukkan secara rinci data yang berhasil atau tidak berhasil diidentifikasi dengan benar. Nilai 1 menyatakan berhasil diidentifikasi dengan benar dan nilai 0 menyatakan tidak dapat diidentifikasi dengan benar. Untuk selengkapnya hasil pengujian dapat dilihat pada Lampiran 3 dan detail tiap pengujian pada Lampiran 4.

Dari tabel hasil pengujian pada Lampiran 3 terlihat bahwa dengan laju pembelajaran yang semakin besar, jaringan akan konvergen lebih cepat. Laju pembelajaran bertindak sebagai *step size* sehingga besar kecilnya laju pembelajaran harus diperhatikan. Laju pembelajaran yang kecil akan membuat jaringan membutuhkan waktu yang lama dalam mencapai kekonvergenan, sedangkan jika terlalu besar dapat membuat jaringan tidak

dapat mencapai apa yang diinginkan. Dari 3 laju pembelajaran yang diujikan yaitu 0,1; 0,2 dan 0,3 terlihat pada nilai 0,1 jaringan membutuhkan *epoch* yang lebih banyak dari nilai 0,2 dan 0,3 tetapi nilainya tidak terlalu jauh berbeda. Hal ini lebih dikarenakan jumlah neuron *input* yang dapat dikatakan sedikit, yaitu hanya 64 buah. Penetapan laju pembelajaran ini akan lebih berpengaruh pada jaringan yang mempunyai jumlah neuron yang besar. Pada nilai 0,2 dan 0,3 terjadi penurunan jumlah *epoch* tetapi generalisasi yang dihasilkan tidak terlalu jauh berbeda. Hal ini berarti jaringan tetap memiliki kinerja yang tidak jauh berbeda dengan nilai laju pembelajaran sebelumnya, tetapi dicapai dalam *epoch* yang lebih sedikit.

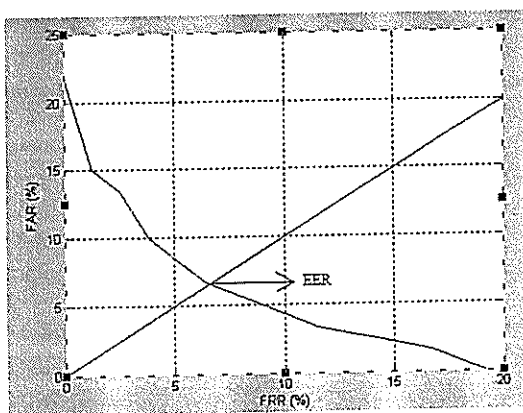
Tabel 4. Hasil percobaan pertama identifikasi pembicara

Pembicara	Data 1	Data 2	Data 3	Data 4	Data 5
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	0	1	1	1
5	1	0	1	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	1	1	0	1	1
9	1	1	1	1	1
10	1	1	1	1	1
11	1	0	0	1	1
12	1	0	1	1	1
13	0	1	1	1	1

Dari 3 toleransi galat yang dicobakan yaitu 0,01; 0,001 dan 0,0001 terlihat bahwa jaringan memiliki generalisasi yang relatif lebih besar pada toleransi galat yang lebih kecil. Pada nilai toleransi galat antara 0,001 dan 0,0001 hanya terjadi sedikit peningkatan generalisasi. Hal ini berarti jaringan mulai terlalu spesifik mengenali pola-pola pembelajaran. Toleransi galat 0,0001 dapat dikatakan merupakan pilihan terbaik karena nilai generalisasi yang dihasilkan lebih besar dari yang lain walaupun membutuhkan waktu dan jumlah *epoch* yang lebih lama untuk konvergen, tetapi besarnya perbedaan waktu dan jumlah *epoch* yang diperlukan tidaklah terlalu signifikan. Pelatihan dengan lebih memperkecil toleransi galat dikhawatirkan akan menuju pada jaringan yang terlalu spesifik mengenali pola tertentu (*overtrained*), sehingga kemampuannya untuk mengenali pola yang serupa (*unseen data*) akan menurun dan hanya dapat mengenali pola-pola pembelajaran (*seen data*).

Verifikasi

Setelah proses identifikasi selesai dilakukan maka proses penentuan *threshold* dapat dilakukan. Konfigurasi JST yang dianggap terbaik digunakan untuk melakukan pengujian verifikasi. Konfigurasi yang dipilih adalah toleransi galat 0,0001 dan laju pembelajaran 0,3. Setelah vektor m yang terurut terbentuk, nilai *threshold* awal yang dipilih adalah elemen pertama dari tiap vektor m . Pemilihan dengan cara ini berarti dipilih nilai *threshold* yang terkecil dan nilai FRR yang dihasilkan akan sama dengan 0%. Dari pengujian ternyata didapat nilai FAR sebesar 19,2308%. *Threshold* dapat dinaikkan untuk pembicara yang memiliki *false acceptance*. *Threshold* diambil dari vektor m dan setiap nilai dari vektor m merepresentasikan besar skor bagi tiap data yang berhasil diidentifikasi dengan benar. Jika dilakukan penaikan *threshold* maka jumlah *false rejection* akan bertambah satu data karena *threshold* awal telah berada di bawah *threshold* saat ini. Penyesuaian nilai *threshold* akan mempengaruhi FAR dan FRR dengan semakin besar nilai FAR maka semakin kecil nilai FRR dan sebaliknya. Setelah dilakukan percobaan dipilih dua nilai FAR dan FRR yang berurutan dan dianggap memberikan keseimbangan bagi keduanya yaitu FAR = 8,9744% dengan FRR = 5% dan FAR = 6,4103% dengan FRR = 6,6667%. Dari dua nilai FAR dan FRR di atas dapat dihitung nilai EER dan didapat nilai EER sebesar 6,5657%. Dari nilai-nilai hasil percobaan dapat dibentuk grafik DET seperti pada Gambar 10.



Gambar 10. Grafik *detection error tradeoff* (DET).

Grafik DET dapat dianalisis lebih lanjut untuk menilai kinerja sistem dan untuk menentukan *threshold* yang sebaiknya digunakan. Sistem dengan tingkat keamanan yang tinggi akan

memilih nilai FAR yang kecil walaupun dengan konsekuensi nilai FRR yang besar (berada pada daerah sebelah kanan bawah grafik DET). Sistem yang memilih keseimbangan akan beroperasi pada daerah sekitar EER dan sistem yang memilih kenyamanan akan memilih FRR yang kecil walaupun diperoleh FAR yang besar. Sistem yang memilih kenyamanan biasanya bertujuan agar pengguna sistem tidak perlu berusaha terlalu keras agar dapat diterima oleh sistem dan lebih toleran terhadap *false acceptance*.

Kelebihan dan Keterbatasan Sistem

Kelebihan sistem. Sistem dapat melakukan pengenalan pembicara yang terbagi dalam dua bagian yaitu identifikasi pembicara dan verifikasi pembicara. Dengan pemilihan besaran peubah yang baik (dalam penelitian ini pada konfigurasi laju pembelajaran 0,2 dan toleransi galat 0,001 dan pada toleransi galat 0,3 dan toleransi galat 0,0001) sistem dapat mengidentifikasi dengan generalisasi tertinggi 92,3077% dan melakukan verifikasi dengan EER 6,5657%.

Pemilihan nilai peubah dapat dilakukan dengan lebih mudah karena sistem dibangun dengan *graphical user interface* (GUI) (contoh tampilan program dapat dilihat pada Lampiran 5). Sistem juga menyediakan fasilitas untuk melakukan identifikasi dan verifikasi secara *real time*.

Keterbatasan sistem. Sistem masih mempunyai keterbatasan dalam melakukan pengenalan. Beberapa hal yang menjadi keterbatasan sistem antara lain:

- Lamanya waktu yang dibutuhkan untuk melakukan PCA, yaitu sekitar 20 menit.
- Belum adanya fasilitas untuk melakukan penambahan pembicara secara otomatis.
- JST *backpropagation* tidak bersifat *incremental learning*, yaitu tidak dapat mengenal pola baru dan jika ingin dilakukan penambahan pembicara maka pembelajaran harus diulang dengan mengikutsertakan pembicara (pola) baru. PCA juga harus dilakukan kembali karena matriks transformasi yang dihasilkan hanya berlaku bagi pola pembelajaran. Hal ini mengakibatkan lamanya waktu yang dibutuhkan untuk melakukan penambahan pembicara.

Selain itu terdapat beberapa masalah klasik yang selalu ditemui dalam suatu sistem pengenalan pembicara, yaitu:

- *Intraspeaker variability*. Variasi pengucapan suara oleh pengguna yang dapat terjadi karena perbedaan keadaan emosi atau fisik. Hal ini sedikit banyak dapat diatasi dengan memperbanyak jumlah *sample* untuk pembelajaran atau dengan meningkatkan variasi *sample* pembelajaran.
- Kualitas alat audio. Penggunaan alat audio yang lebih baik diharapkan dapat memberikan kinerja yang lebih baik melalui peningkatan kualitas suara.
- Keadaan lingkungan. Lingkungan dengan *noise* yang besar akan menurunkan kinerja sistem.
- *Impostor*. *Impostor* dalam hal ini dapat dibagi menjadi 2 bagian yaitu pengguna lain yang berusaha menirukan kata-kata pengguna sebenarnya dan penggunaan media perekaman (*tape recorder*). Salah satu cara yang telah diterapkan untuk mengatasi *impostor* adalah dengan memperbesar *threshold* untuk meminimumkan FAR.

KESIMPULAN DAN SARAN

Kesimpulan

JST *backpropagation* dapat melakukan pembelajaran dan pengenalan terhadap suatu pola dengan tingkat generalisasi yang cukup tinggi. Identifikasi dengan generalisasi tertinggi (92,3077%) dicapai pada toleransi galat 0,001 dengan laju pembelajaran 0,2 dan toleransi galat 0,0001 dengan laju pembelajaran 0,3. Sistem verifikasi menghasilkan nilai EER sebesar 6,5657% yaitu nilai yang memberikan keseimbangan antara FAR dan FRR ($FAR \approx FRR$).

Semakin kecil laju pembelajaran JST *backpropagation* maka semakin banyak *epoch* yang dibutuhkan untuk mencapai konvergensi. Pemilihan laju pembelajaran yang baik akan menghasilkan generalisasi yang baik dengan jumlah *epoch* yang kecil.

Penurunan nilai toleransi galat terbukti meningkatkan generalisasi, tetapi pada suatu saat penurunan nilai ini tidak lagi berpengaruh secara signifikan pada generalisasi. Pada saat tersebut penurunan kembali toleransi galat sebaiknya tidak dilakukan karena akan menuju pada jaringan yang *overtrained* dan akan terlalu spesifik mengenali suatu pola tertentu.

Proses *feature extraction* dan *feature selection* mampu mengekstrak data suara yang berdimensi besar menjadi data baru yang berdimensi kecil yang merepresentasikan data asli. Proses-proses sinyal digital seperti *frame blocking*, *frame windowing* dan analisis *cepstral* dapat mengekstrak pola suara yang dihasilkan. Hasil ekstraksi berupa vektor *feature* dapat diolah lebih lanjut dengan teknik *feature selection* seperti PCA. Dengan PCA dapat dibentuk data yang tereduksi tanpa harus kehilangan banyak informasi.

Kombinasi teknik-teknik tertentu dibutuhkan dalam membangun suatu sistem pengenalan pembicara. Dalam hal ini proses-proses sinyal digital (*frame blocking*, *frame windowing* dan analisis *cepstral*), teknik *feature selection* dengan PCA dan *pattern matching* dengan JST *backpropagation* dapat melakukan hal tersebut.

Saran

Penelitian ini masih dapat dikembangkan lebih jauh dan lebih dalam lagi yang nantinya diharapkan dapat terbentuk suatu sistem yang lebih baik. Saran-saran bagi penelitian ini lebih lanjut antara lain:

- Penggunaan teknik pemrosesan sinyal digital yang lain seperti *vector quantization* dan *linear predictive analysis* untuk kemudian dibandingkan hasilnya sehingga dapat ditentukan teknik yang paling optimal.
- Penggunaan teknik *feature selection* selain PCA, seperti *linear discriminant analysis* dan *independent component analysis* untuk kemudian juga dibandingkan sehingga diperoleh teknik yang paling optimal.
- Penggunaan JST yang bersifat *incremental learning* sehingga JST dapat mengenali pola baru dengan lebih cepat.
- Penggunaan teknik *pattern matching* selain JST atau mengkombinasikannya dengan JST bila memungkinkan (membentuk suatu teknik hibrida yang diharapkan memberikan hasil yang lebih baik).
- Penggunaan teknik *filtering*, *noise reduction*, dan *end point detection* sehingga sinyal suara digital yang dihasilkan akan lebih baik dari segi kualitas maupun dalam jumlah besarnya data.
- Melakukan penambahan jumlah pembicara untuk melihat kinerja sistem dengan jumlah data yang besar.

- Tersedianya fasilitas yang memungkinkan penambahan pembicara secara otomatis.
- Penggunaan alat-alat audio (mikrofon dan kartu suara) yang lebih baik sehingga data audio yang diperoleh akan lebih baik kualitasnya.

DAFTAR PUSTAKA

- Campbell, J.P, JR. 1997. Speaker Recognition: A Tutorial. Proc. IEEE, vol. 85, no. 9, pp. 1437-1462, 1997.
- Fausett, L. 1994. Fundamentals of Neural Network. Prentice Hall, Englewood Cliffs, NJ.
- Fu, L. 1994. Neural Network in Computer Intelligence. McGraw-Hill, Singapore.
- Herryadie, F.D. 1999. Penggunaan Analisis Komponen Utama dan Jaringan Syaraf Propagasi Balik untuk Pengenalan Wajah. Skripsi. Jurusan Ilmu Komputer, IPB.
- Ho, C.E. 1998. Speaker Recognition System. Project Report. California Institute of Technology.
- Owens, F.J. 1993. *Signal Processing of Speech*. Macmillan, London.
- Riadi. 2001. Jaringan Syaraf Tiruan untuk Pengenalan Tanda Tangan. Skripsi. Jurusan Ilmu Komputer, IPB.
- Roweis, S. 1998. Speech Processing Background. http://www.dna.caltech.edu/~courses/cns187/references/roweis_spblet.ps. [16 Maret 2004].
- Shlens, J. 2003. A Tutorial on Principal Component Analysis. <http://www.sn1.salk.edu/~shlens/pub/notes/pca.pdf>. [16 Maret 2004].
- Xafopoulos, A. 2001. Speaker Verification (an overview). TUT – TICSP presentation. TICSP (Tampere International Center for Signal Processing), TUT (Tampere Univ. of Technology), Tampere, Finland.

LAMPIRAN

Lampiran 1. Algoritme pembelajaran JST *backpropagation*

- Inisialisasi bobot
Set semua bobot dan *threshold* tiap *node* dengan angka acak yang kecil. Selain dengan nilai acak, inisialisasi bobot juga dapat dilakukan dengan inisialisasi Nguyen-Widrow. Dengan inisialisasi Nguyen-Widrow diharapkan proses pembelajaran yang dilakukan jaringan menjadi lebih singkat. Inisialisasi Nguyen-Widrow didefinisikan sebagai berikut:
 n : Jumlah unit *input*
 p : Jumlah neuron lapisan tersembunyi
 β : Faktor pengali
 $\beta = 0,7\sqrt{p}$
 Untuk setiap unit tersembunyi ($j=1 \dots p$):
 - Inisialisasi v_{ij} (lama) = bilangan acak antara -0,5 dan 0,5 (atau antara $-\gamma$ dan γ).
 - Hitung $\|v_{ij} \text{ (lama)}\|$
 - Inisialisasi ulang bobot

$$v_{ij} = \frac{\beta v_{ij} \text{ (lama)}}{\|v_{ij} \text{ (lama)}\|}$$
 - Set bias v_{oj} = bilangan acak antara $-\beta$ dan β .
 - Perhitungan nilai pengaktifan
Level pengaktifan (O_j) dari unit tersembunyi dan unit *output* ditentukan sebagai berikut:

$$O_j = F(\sum W_{ji} O_i - \theta_j)$$
 dengan W_{ji} adalah bobot dari *input* O_i , θ_j adalah *threshold* dari unit j , dan F adalah fungsi pengaktifan.
 - Penyesuaian bobot
Penyesuaian bobot dilakukan secara rekursif dimulai dari unit *output* berlanjut ke unit tersembunyi dan dilakukan dengan cara sebagai berikut:

$$W_{ji}(t+1) = W_{ji}(t) + \Delta W_{ji}$$
 dengan $W_{ji}(t)$ adalah bobot dari unit i ke unit j pada waktu t (iterasi ke t) dan ΔW_{ji} adalah penyesuaian bobot. ΔW_{ji} dihitung melalui persamaan berikut:

$$\Delta W_{ji} = \eta \delta_j O_i$$
 dengan η adalah laju pembelajaran independen yang dicobakan ($0 < \eta < 1$, misal 0,3) dan δ_j adalah gradien *error* pada unit j . Gradien *error* didapat dari persamaan sebagai berikut:
 - Untuk unit *output*:

$$\delta_j = O_j(1 - O_j)(T_j - O_j)$$
 dengan T_j adalah *output* pengaktifan yang diinginkan dan O_j adalah *output* pengaktifan yang dihasilkan oleh unit j .
 - Untuk unit tersembunyi:

$$\delta_j = O_j(1 - O_j) \sum_k \delta_k W_{kj}$$
 dengan δ_k adalah gradien *error* pada unit k , dan unit tersembunyi j koneksinya terarah pada unit k .
- Iterasi terus dilakukan sampai konvergen. Konvergen dicapai saat kriteria *error* tertentu dipenuhi. Iterasi akan melibatkan memberikan *instance*, menghitung pengaktifan dan memodifikasi bobot.

Lampiran 2. Tabel percobaan jumlah neuron tersembunyi dengan tiga kali pengulangan pada toleransi galat 0,001 dan laju pembelajaran 0,1

Jumlah Neuron Tersembunyi	Ulangan Pembelajaran Ke-	Epoch	Waktu Pembelajaran (detik)	Diidentifikasi dengan benar	Generalisasi (%)	Waktu Pengujian (detik)
10	1	1260	18,068	52/65	80	0,016
	2	1201	17,047	53/65	81,5385	0
	3	2386	38,25	54/65	83,0679	0,015
20	1	175	2,484	57/65	87,6923	0,015
	2	153	1,969	48/65	73,8462	0
	3	186	2,391	54/65	83,0769	0,016
30	1	127	1,769	58/65	89,2308	0,016
	2	138	2	55/65	84,6154	0,016
	3	137	1,922	58/65	89,2308	0,016
40	1	119	2,109	56/65	86,1538	0,016
	2	126	1,953	57/65	87,6923	0,015
	3	121	1,75	56/65	86,1538	0,016
50	1	119	1,922	56/65	86,1538	0,015
	2	119	1,938	54/65	83,0769	0,016
	3	128	1,75	56/65	86,1538	0,016
60	1	116	1,875	54/65	83,0769	0,016
	2	116	1,969	56/65	86,1538	0,016
	3	117	2,094	58/65	89,2308	0,016
70	1	114	2,141	59/65	90,7692	0,015
	2	115	2,172	57/65	87,6923	0,015
	3	113	2,016	59/65	90,7692	0,016
80	1	111	2,187	58/65	89,2308	0,016
	2	113	2,063	59/65	90,7692	0,016
	3	113	2,219	58/65	89,2308	0,016
90	1	111	2,704	58/65	89,2308	0,015
	2	112	2,578	57/65	87,6923	0,015
	3	110	2,484	60/65	92,3077	0,016
100	1	111	2,844	59/65	90,7692	0,015
	2	111	3,172	59/65	90,7692	0,015
	3	111	2,765	58/65	89,2308	0,015

Lampiran 3. Tabel hasil pengujian identifikasi

PEMBELAJARAN					PENGUJIAN		
Toleransi Galat	Laju Pembelajaran	Ulangan Pembelajaran Ke-	Epoch	Waktu Pembelajaran (detik)	Diidentifikasi dengan benar	Generalisasi (%)	Waktu Pengujian (detik)
0,01	0,3	1	56	1,297	53/65	81,5385	0,015
		2	51	1,203	53/65	81,5385	0,015
		3	53	1,312	56/65	86,1538	0,016
		4	48	1,172	56/65	86,1538	0,016
		5	55	1,265	56/65	86,1538	0,016
	0,2	1	59	1,36	50/65	76,9231	0,032
		2	58	1,343	58/65	89,2308	0,015
		3	59	1,391	54/65	83,0769	0,016
		4	56	1,281	50/65	76,9231	0,015
		5	61	1,469	57/65	87,6923	0,016
	0,1	1	74	1,985	56/65	86,1538	0,125
		2	75	1,813	54/65	83,0769	0,016
		3	74	1,672	52/65	80	0,016
		4	80	1,859	55/65	84,6154	0,016
		5	77	1,782	54/65	83,0769	0,015
0,001	0,3	1	86	1,953	55/65	84,6154	0,016
		2	88	2,016	59/65	90,7692	0,016
		3	88	1,75	56/65	86,1538	0,016
		4	90	2,14	56/65	86,1538	0,016
		5	88	2	55/65	84,6154	0,016
	0,2	1	99	2,266	59/65	90,7692	0,016
		2	124	3,438	57/65	87,6923	0,016
		3	97	2,203	57/65	87,6923	0,016
		4	100	2,203	60/65	92,3077	0,016
		5	96	2,125	58/65	89,2308	0,016
	0,1	1	109	2,453	58/65	89,2308	0,016
		2	110	2,469	57/65	87,6923	0,016
		3	108	2,484	58/65	89,2308	0,016
		4	110	2,39	59/65	90,7692	0,015
		5	111	2,578	59/65	90,7692	0,016
0,0001	0,3	1	137	3,156	58/65	89,2308	0,016
		2	133	3,063	58/65	89,2308	0,015
		3	136	3,156	58/65	89,2308	0,016
		4	135	3,047	60/65	92,3077	0,016
		5	136	3,125	58/65	89,2308	0,015
	0,2	1	144	3,219	56/65	86,1538	0,016
		2	143	2,891	56/65	86,1538	0,016
		3	144	3,156	58/65	89,2308	0,016
		4	142	3,094	58/65	89,2308	0,016
		5	144	3,234	58/65	89,2308	0,015
	0,1	1	156	3,562	59/65	90,7692	0,016
		2	157	3,547	59/65	90,7692	0,016
		3	157	3,5	59/65	90,7692	0,016
		4	159	3,578	59/65	90,7692	0,015
		5	157	3,5	57/65	87,6923	0,016

Lampiran 4. Tabel hasil identifikasi secara terperinci

Toleransi Galat	Laju Pembelajaran	Ulangan Pembelajaran Ke-	Pembicara 1	Pembicara 2	Pembicara 3	Pembicara 4	Pembicara 5	Pembicara 6	Pembicara 7	Pembicara 8	Pembicara 9	Pembicara 10	Pembicara 11	Pembicara 12	Pembicara 13
0,01	0,1	1				1011	10100			11011			10011	10111	01111
		2		01100		10011	01110			11011		01111	10111	10111	01111
		3		01010		10010	10100							10011	00111
		4		11010		10010	10100					01111		10111	
		5		11010		10010	10100							10011	01111
	0,2	1		01000		10011	10100			11010		01111	11011	10011	
		2		11110			10100			11110				10111	01111
		3		11100		10110	10100		01111			01111		10111	01111
		4				10011	10100		01111	11010		01111		00011	00101
		5		11011			10100		01111					10011	01111
	0,3	1				10011	10100					01111		10111	00000
		2				10111	10100		11110	01110		01111		10011	11100
		3				10011	10100			11100				10011	
		4		01001			10110	10111	11110					10011	
		5		11110		10111	10100			11011				10111	01011
0,001	0,1	1		11110		10011	10100						01111		
		2				10011	10100							00111	
		3				10111	10100					01111		00111	
		4				10011	10100							10111	
		5		11110			10100		11110					10111	
	0,2	1				11011	10100							10111	01111
		2		11110		10111	10100					01111		10111	01111
		3				10011	10100			11110		01111		10111	
		4				10111	10110		01111					10111	
		5		11110		10011	10100	10111				01111		10111	01111
	0,3	1				10011	10100					01111		10111	
		2				10111	10110			11110				10111	
		3		11010		10111	10100		01111			01111		10111	
		4				10011	10100		01111					10011	
		5				10111	10100		01111	11010		01111		00111	



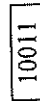
Lampiran 4. (lanjutan)

Toleransi Galat	Laju Pembelajaran	Ulangan Pembelajaran Ke-	Pembicara 1	Pembicara 2	Pembicara 3	Pembicara 4	Pembicara 5	Pembicara 6	Pembicara 7	Pembicara 8	Pembicara 9	Pembicara 10	Pembicara 11	Pembicara 12	Pembicara 13
0,0001	0,1	1				10111	10100					01111		10111	
		2				10111	10100					01111		10111	
		3					10100					01111		10111	01111
		4				10111	10100					01111		10111	
		5				10011	10100					01111		10111	01111
	0,2	1		11110		10111	10100			11110				00111	01111
		2		11110		10111	10100					01111	11011	10011	
		3		11110			10110					01111		00111	01111
		4				10111	10100		01111					10111	01111
		5		11110		10111	10100							10111	01111
	0,3	1				10111	10100			11011				10111	01111
		2				10111	10100			11110				00111	
		3				10011	10100					01111		10111	
		4				10111	10100							10111	
		5				10111	10100			11110				10111	01111

Keterangan:



Seluruh 5 sample pengujian berhasil diidentifikasi dengan benar



Dari 5 sample, sample ke-2 dan ke-3 tidak berhasil diidentifikasi dengan benar

Lampiran 5. Contoh tampilan program

Speaker Recognition

Training

Learning rate	0.3
Error	0.0001
Principal Component	64
Hidden Neuron	100
<hr/>	
<input type="button" value="Train"/>	Elapsed time 3.39
<input type="button" value="Reset"/>	Epochs 133

Testing

Elapsed time	0.015
Generalization	92.3077 %

Identification

Your Identity is **Baskoro O**

Testing

Threshold		FAR	7.6923 %
Speaker	10	FRR	1.6667 %
th	0.866		

Verification

Anda adalah Baskoro O. Verifikasi diterima