PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN MENGGUNAKAN JARINGAN SYARAF TIRUAN FUNGSI BASIS RADIAL

TUGAS AKHIR

Oleh :

F.X. WISDARMANTO ERLANGGA

5196.100.047

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

2002

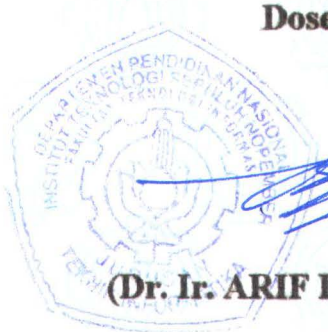
PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN MENGGUNAKAN JARINGAN SYARAF TIRUAN FUNGSI BASIS RADIAL

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer
Pada
Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui,

Dosen Pembimbing,



(Dr. Ir. ARIF DJUNAIDY, M.Sc., Ph.D.)

SURABAYA
Februari, 2002

And in Life's noisiest hour,
There whisper still the ceaseless Love of Thee,
The heart's Self-solace and soliloquy.
You mould my Hopes, you fashion me within ;
And to the leading Love-throb in the Heart
Thro'all my Being, thro' my pulse's beat ;
You lie in all my many Thoughts, like Light,
Like the fair light of Dawn, or summer Eve
On rippling Stream, or cloud-reflecting Lake.
And looking to the Heaven, that bends above you,
How oft! I bless the Lot that made me love you.

ABSTRAK

ABSTRAK

Pengenalan wajah merupakan pekerjaan yang sulit dikarenakan pada proses pembentukan citra terdapat bermacam-macam variabel, seperti kualitas dan pencahayaan citra, geometri, perubahan, dan penyamaran citra. Kebanyakan sistem pemroses citra wajah yang terdapat saat ini hanya dapat menangani basis data citra dengan batasan berupa ukuran citra, umur, jenis kelamin, dan/atau ras. Jaringan syaraf tiruan Fungsi Basis Radial (*Radial Basis Function)* memiliki beberapa kelebihan jika diterapkan sebagai pengklasifikasi dari sistem pengenalan wajah yaitu pada tahap pembelajaran yang diawasi proses konvergen dapat dicapai dengan cepat, mempunyai kemampuan untuk mengelompokkan citra-citra serupa sebelum mengklasifikasikannya dan kemampuan mengenali citra wajah lebih dari satu orang, tanpa memandang berbagai ekspresi wajah yang ada.

Dalam tugas akhir ini dibuat sistem pengenalan wajah yang dimulai dengan mendeteksi sebuah citra dengan mencari bagian mana yang merupakan wajah, lalu menandainya dengan sebuah kotak, kemudian dilanjutkan dengan menormalisasi citra wajah berdasarkan geometri dan perubahan pencahayaan berdasarkan kotak yang menandai wajah/lokasi mata. Citra ternormalisasi tersebut kemudian direpresentasikan menjadi sebuah vektor linear yang berisi urutan piksel dari citra tersebut. Dengan menggunakan algoritma *k-means clustering,* pada tahap pelatihan tidak terawasi, data pelatihan yang berisi beberapa vektor linear dikelompokkan berdasarkan kemiripannya. Hasil pelatihan yang didapat dari sistem merupakan hasil perhitungan fungsi basis radial gaussian pada layer *hidden* yang terdiri dari bobot, mean, varian dari tiap-tiap unit *hidden* dan jumlah unit *hidden.* Setelah parameter tersebut didapat maka proses pengenalan wajah dapat dilakukan.

Dengan penggunaan data pelatihan yang telah dipilih berdasarkan variasi data yang mewakilinya, perangkat lunak pengenalan wajah yang telah berhasil dibuat dapat memberikan prosentase keberhasilan pengenalan sebesar 97,84% dengan waktu komputasi untuk proses pelatihan, untuk 185 citra milik 37 macam individu, hanya memerlukan waktu kurang dari 5 menit dan proses pengenalan hanya memerlukan waktu tidak lebih dari 5 detik dari berbagai macam citra dengan ekspresi wajah yang ada.

KATA PENGANTAR

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Pengasih, karena berkat rahmat-Nya penulis dapat menyelesaikan pembuatan Tugas Akhir yang berjudul:

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN MENGGUNAKAN JARINGAN SYARAF TIRUAN FUNGSI BASIS RADIAL

Selama beberapa bulan dalam proses pembuatan Tugas Akhir ini, penulis mengalami berbagai macam kesulitan dan hambatan yang cukup melelahkan. Namun berkat dukungan, dorongan dan semangat dari keluarga, teman dan bapak/ibu dosen, akhirnya Tugas Akhir ini dapat diselesaikan. Untuk itu penulis hendak mengucapkan terima kasih sebesar-besarnya kepada:

1. **Tuhan Sang Pencipta**, atas berkat dan rahmat-Nya dan atas perlindungannya sampai penulis menginjak usia sekarang.
2. Ayahanda dan ibunda tercinta **Teguh Erlangga** dan **Moedinar Poedjaningsih** yang telah merawatku semenjak kecil hingga sekarang dan atas dorongan baik moral dan spiritual yang tak pernah henti-hentinya diberikan kepada penulis.
3. Kakak-kakakku mbak **Ririk**, mbak **Antik** dan mbak **Hedy** yang selalu menanyakan kapan penulis lulus dan atas segala bantuan dan semangat yang diberikan.

4. **Dr. Ir. Arif Djunaidy, M.Sc., Ph.D.**, selaku dekan dan pembimbing atas segala bimbingan dan pengarahan selama ini.
5. **Rully Soelaiman, S.Kom**, yang telah menyumbangkan ide, saran dan paper yang merupakan awal dari Tugas Akhir ini.
6. **Dr. Ir. Riyanarto Sarno**, selaku dosen wali atas bimbingan, bantuan dan pengarahan selama ini.
7. Seluruh Staff dosen dan karyawan Teknik Informatika ITS atas segala bantuan selama penulis menjalani masa-masa perkuliahan.
8. Teman setiaku **Emi**, atas semangat dan perhatian yang tak pernah henti juga atas dorongan yang begitu besar bagi penulis untuk menyelesaikan Tugas Akhir ini.
9. **Yushar Yahya**, atas sumbangan otak cemerlangnya yang menjadi kunci pembuka masalah –masalah yang dihadapi penulis.
10. Seluruh **kru AJK**: Tono, Wayan, Nunut, "Gimbal", Wiro, Ferry, "Damen", Guruh, Dwi, Bornok, Ujik, Eddy, Ronie, "lemot kecil" dan yang tidak bisa penulis sebutkan satu-satu, terima kasih atas kebersamaan kita selama ini.
11. **Suyatno**, **Wiyono**, **Ismanu**, Mas **Yudi**, Mas **Soleh**, Mas **Sugeng**, Pak **Supriyo**, Pak **Purnomo**, yang selalu menunggu dan menagih janji apabila penulis telah menyelesaikan Tugas Akhir.
12. Seluruh rekan-rekan C0C, **Danar**, **Purno**, **Monica**, **Diana** atas KP-nya yang indah, **Ninis** atas tumpangan tempat tinggal selama penulis menjalani masa kuliah, dan semuanya yang tidak bisa penulis sebutkan satu-persatu, semoga

motto angkatan tidak pernah pupus "HANYA MAKAN-MAKAN YANG MENYATUKAN KITA"

13. Seluruh penghuni Wisma Permai Tengah ii/31 atas kenangan yang menyenangkan diawal-awal semester kuliah.
14. Teman-teman kampung halamanku, **Mufti**, **Onil**, "**Manol**", atas dukungan dan semangat disaat penulis mengalami masa-masa stres.
15. Semua pihak baik secara langsung maupun tidak langsung membantu kelancaran pembuatan tugas akhir ini.

Saya menyadari Tugas Akhir ini masih jauh dari sempurna dan banyak kekurangannya. Kritik dan saran saya harapkan demi sempurnanya Tugas Akhir ini. Semoga Tugas Akhir ini dapat bermanfaat bagi semua pihak yang memerlukannya.

Surabaya, Februari 2002

Penulis

DAFTAR ISI

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR TABEL

BAB I

PENDAHULUAN

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Pengenalan wajah kebanyakan merupakan pekerjaan yang sulit dikarenakan pada proses pembentukan citra mempunyai bermacam-macam variabel seperti: kualitas dan pencahayaan citra, geometri, perubahan, dan penyamaran citra.[SRI-97] Kebanyakan sistem pemroses citra wajah yang terdapat saat ini hanya dapat menangani citra wajah dengan batasan: ukuran citra, umur, jenis kelamin, atau ras dan selanjutnya sistem tersebut memberikan kontrol dan pengawasan terhadap faktor atau kondisi yang mempengaruhi sistem tersebut. Ada beberapa tambahan tingkatan variabel mulai dari sistem yang mengasumsi bahwa posisi wajah dan kondisi yang mempengaruhinya (jarak dan pencahayaan) sepenuhnya diawasi, ke sistem yang melibatkan sedikit atau tidak sama sekali kontrol terhadap latar belakang dan sudut pandang dan sampai ke sistem yang mengijinkan perubahan besar pada penampakan wajah dengan beberapa faktor seperti usia dan penyamaran (memakai topi atau kacamata).

Tugas akhir ini mempelajari arsitektur klasifikasi Fungsi Basis Radial dan memperlihatkan kemampuan mengenali citra wajah lebih dari satu orang, tanpa memandang berbagai ekspresi wajah yang ada.

Keuntungan yang diperoleh dari penggunaan jaringan syaraf tiruan Fungsi Basis Radial :

1. Pada tahap pembelajaran yang diawasi proses konvergen dapat dicapai dengan cepat karena hanya terdapat satu layer saja yang terlibat di dalamnya.
2. Penggunaan aproksimasi yang jauh lebih baik daripada interpolasi dalam menangani 'noisy' data khususnya data yang berupa citra wajah.[JON-96]
3. Kemampuannya untuk mengelompokkan citra-citra serupa sebelum mengklasifikasikannya.[JON-96]

1.2 PERMASALAHAN

Wajah manusia mempunyai derajat kemiripan yang tinggi antara yang satu dengan yang lainnya. Hanya dari satu individu sudah bisa menghasilkan berbagai macam citra wajahnya menurut ekspresi muka, tatanan rambut, sudut pandang, pencahayaan, dan lain-lain. Permasalahan yang timbul adalah

1. Bagaimana sistem yang akan dibuat dapat mengatasi hal tersebut mengingat pada sistem tersebut nantinya akan dipakai untuk mengenali lebih dari satu individu yang mempunyai citra wajah yang bermacam-macam..
2. Bagaimana menerapkan sistem tersebut menjadi suatu perangkat lunak pengenalan wajah manusia yang dapat melakukan proses pengenalan citra wajah manusia dengan besarnya nilai kesalahan pengenalan seminimal mungkin.

1.3 TUJUAN DAN MANFAAT

Tujuan dari penulisan tugas akhir ini adalah membuat suatu sistem pengenalan wajah manusia dengan menggunakan sistem jaringan syaraf tiruan Fungsi Basis Radial yang dapat memanfaatkan kelebihan-kelebihan yang menjadi pertimbangan dipakainya metoda ini dan mengatasi permasalahan yang terdapat pada proses pengklasifikasi dalam sistem pengenalan wajah.

1.4 BATASAN PERMASALAHAN

Pada tugas akhir ini hanya membahas rancangan perangkat lunak yang dikembangkan untuk mewujudkan dan mengimplementasikan proses pengenalan wajah pada tingkat dimana pekerjaan pengklasifikasian diperlukan.

1. Model jaringan syaraf tiruan yang digunakan adalah Fungsi Basis Radial (FBR).
2. Citra/gambar yang digunakan adalah citra *gray-scale* bertipe bitmap yang yang didapat oleh laboratorium riset *Olivetti* dengan ukuran 112x92 piksel.
3. Diasumsikan citra nomalisasi, berukuran 68x68 piksel, didapat dari proses sebelumnya yaitu pendeteksi wajah, sehingga pada tugas akhir ini untuk memperoleh citra normalisasi dilakukan secara manual. Dimana pada citra ternomalisasi dapat ditemukan bentuk: sepasang mata, hidung dan mulut. Sedapatnya ketiga ciri wajah tersebut tersusun secara proporsional dalam citra ternomalisasi.

4. Yang dimaksud dengan nilai positif dari proses pendeteksian adalah citra wajah yang merupakan wajah manusia baik menggunakan ataupun tidak menggunakan atribut tertentu seperti memakai kacamata dan citra wajah tersebut dapat diidentifikasikan.

1.5 METODOLOGI PENGERJAAN TUGAS AKHIR

- Studi kepustakaan

  Pencarian sumber-sumber yang berhubungan dengan pengembangan perangkat lunak ini, berupa buku, jurnal dan literatur yang lain sehingga pengerjaan tugas akhir ini dapat berjalan dengan lancar.

- Perancangan dan pembuatan perangkat lunak

  Solusi dari permasalahan dirancang dan dibuat menjadi suatu algoritma yang merupakan kerangka dari perangkat lunak yang akan dikembangkan dan dilanjutkan dengan pembuatan perangkat lunak.

- Pengujian perangkat lunak

  Perangkat lunak yang telah dikembangkan ini diuji dengan menggunakan variasi data yang mewakili semua permasalahan yang ada.

- Evaluasi dan modifikasi perangkat lunak

  Evaluasi dan modifikasi ini untuk mengoptimalkan kerja dari perangkat lunak yang telah dibuat sehingga diperoleh hasil yang baik.

- Penulisan tugas akhir

  Membuat laporan dalam bentuk buku Tugas Akhir yang dapat menjelaskan semua proses dari tahap perancangan sampai tahap implementasi berikut

dengan evaluasi yang didapat dari hasil uji coba sehingga dengan membaca buku Tugas Akhir ini dapat mengerti keseluruhan proses pengenalan wajah dengan menggunakan jaringan syaraf tiruan Fungsi BasisRadial.

1.6 SISTEMATIKA PEMBAHASAN

Sistematika yang digunakan dalam Tugas Akhir dijelaskan berikut ini.

- Bab I Pendahuluan, menjelaskan mengenai latar belakang, permasalahan dan batasannya, tujuan dan manfaat, metodologi penelitian dan sistematika penulisan.
- Bab II Jaringan Syaraf Tiruan, berupa pembahasan dasar teori jaringan syaraf tiruan yang secara langsung maupun tidak menjadi referensi dalam mengerjakan tugas akhir ini.
- Bab III Pengenalan Wajah dengan Menggunakan Fungsi Basis Radial, menjelaskan mengenai metoda yang digunakan pada tugas akhir ini untuk proses pengenalan citra wajah manusia..
- Bab IV Perancangan dan Pembuatan Perangkat Lunak, membahas sistem perangkat lunak pengenalan wajah dengan menggunakan jaringan syaraf tiruan model Fungsi Basis Radial, perancangan dan implementasi struktur data yang digunakan oleh perangkat lunak .
- Bab V Uji Coba dan Evaluasi Perangkat Lunak, membahas hasil uji coba perangkat lunak dan mengevaluasi kemampuannya.
- Bab VI Penutup, menguraikan kesimpulan dari bab-bab yang lain serta kemungkinan pengembangan yang berkaitan dengan perangkat lunak ini.

BAB II

DASAR TEORI

BAB II

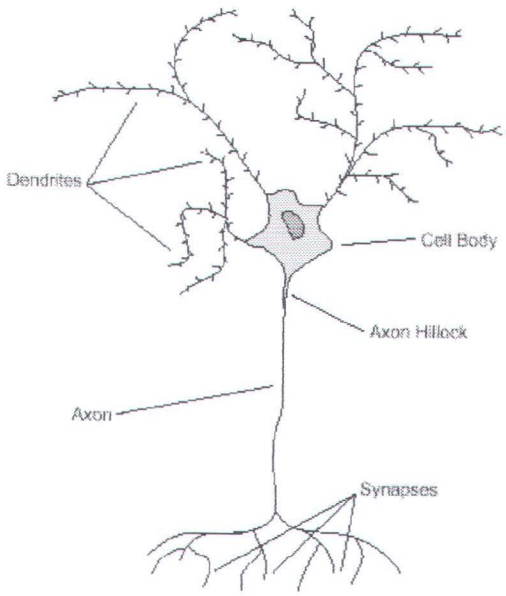
DASAR TEORI

Pekerjaan dalam bidang jaringan syaraf tiruan dimula ketika diketahui bahwa otak bekerja dengan cara yang sangat berbeda dengan komputer digital konvensional saat itu. Otak merupakan komputer pararel, nonlinear yang sangat kompleks. Yang mempunyai kemampuan untuk mengatur neuron untuk melakukan suatu perhitungan komputasi (mis: pengenalan pola, persepsi dan pengontrol motor) yang lebih cepat dari pada komputer tercepat saat ini.

2.1 JARINGAN SYARAF TIRUAN [BAR-92]

Dengan mengetahui bahwa didalam otak manusia terdapat berjuta-juta neuron sebagai penyusun "komputer" tercanggih saat ini (Gambar 2.1), maka para ilmuwan mulai mencontoh model jaringan syaraf yang terdapat di dalamnya. Jaringan syaraf diartikan sebagai prosesor yang terdistribusi secara pararel dengan kemampuan untuk menyimpan pengetahuan yang telah didapat sebelumnya untuk digunakan kembali di masa mendatang. Proses penyimpanan tersebut dapat dinyatakan dalam dua cara:

1. Pengetahuan didapat oleh jaringan dari proses belajar.
2. Kekuatan hubungan antar neuron yang diketahui sebagai "bobot sinapsis" digunakan untuk menyimpan pengetahuan tersebut.



Gambar 2.1 System syaraf

2.2 MODEL NEURON

Sebuah neuron merupakan unit pemroses informasi yang berperan penting dalam kerja syaraf. Gambar 2.1 menunjukkan model dari sebuah neuron. Dari gambar tersebut dapat kita temui tiga penyusun dasar dari sebuah neuron yaitu:

1. Sekumpulan sinapsis, setiap sinapsis mempunyai bobotnya masing-masing. Secara lebih terperinci, sebuah sinyal x_j pada input sinapsis j terhubung ke neuron k dikalikan dengan bobot sinapsis w_{kj}.
2. Sebuah penjumlah / *adder* untuk menjumlah sinyal input, yang telah dipengaruhi oleh bobot dari sinapsis neuron yang berhubungan. Operasi disini menghasilkan nilai '*linear combiner*'.

3. Sebuah fungsi aktivasi untuk membatasi aplitudo output neuron pada batas nilai tertentu. Umumnya besarnya amplitudo output ternomalisasi ini berkisar dari [0,1] atau [-1.1].

Model neuron yang ditunjukkan oleh gambar 2.1 menggunakan nilai *threshold* eksternal θ_k yang mempunyai fungsinya menurunkan nilai input bagi fungsi aktivasi. Dilain pihak, nilai input ini dapat dinaikkan dengan menggunakan nilai eksternal *bias*; bias merupakan nilai negatif dari threshold.

Secara matematika, neuron k dapat dinyatakan dengan pasangan persamaan berikut ini:

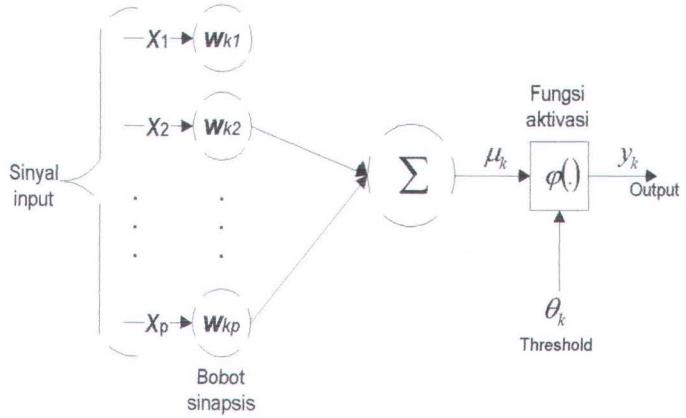
$$\mu_k = \sum_{j=1}^{p} w_{kj} x_j \tag{2.1}$$

dan

$$y_k = \varphi(\mu - \theta_k) \tag{2.2}$$

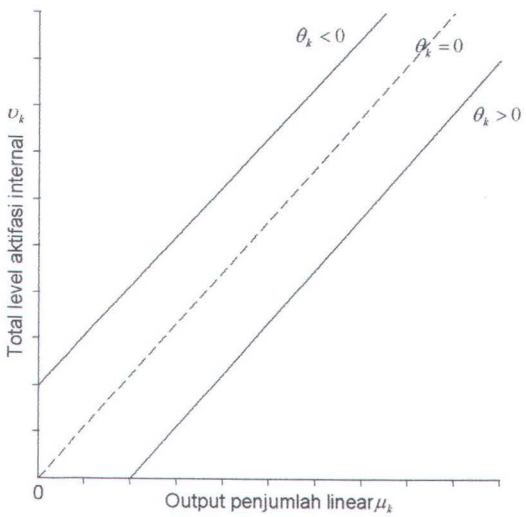
dimana $x_1, x_2, \ldots, x_p$ merupakan sinyal input; $w_{k1}, w_{k2}, \ldots, w_{kp}$ merupakan bobot sinapsis dari neuron k; μ_k merupakan output linear combiner; θ_k sebagai threshold; $\varphi(\cdot)$ merupakan fungsi aktivasi; dan y_k merupakan sinyal output dari neuron. Penggunaan threshold dapat menggeser nilai output μ_k pada model neuron Gambar 2.2, seperti yang ditunjukkan dengan

$$\upsilon_k = \mu_k - \theta_k \tag{2.3}$$



Gambar 2.1 Model nonlinear neuron

Relasi antara level aktivasi internal efektif atau potensi aktivasi υ_k dari neuron k dengan output penjumlah linear μ_k tergantung dari nilai threshold θ_k apakah positif ataukah negatif seperti yang ditunjukkan pada Gambar 2.3.



Gambar 2.2 Pergeseran nilai output dengan adanya threshold

Threshold θ_k merupakan parameter eksternal dari neuron k. Dari persamaan (2.1) dan persamaan (2.2) maka didapatkan

$$\upsilon_k = \sum_{j=0}^{p} w_{kj} x_j \tag{2.4}$$

dan

$$y_k = \varphi(\upsilon_k) \tag{2.5}$$

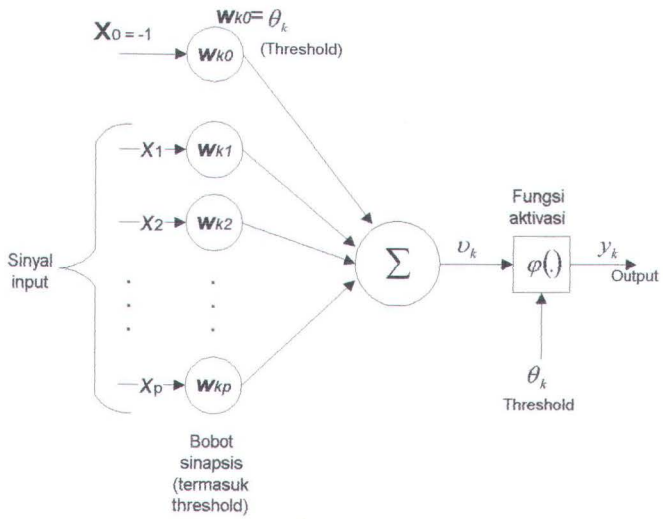
Pada persamaan (2.4) ditambahkan sinapsis baru, yang mempunyai input

$$x_0 = -1 \tag{2.6}$$

dan bobotnya

$$w_{k0} = \theta_k \tag{2.7}$$

Dari persamaan diatas maka dapat dibangun kembali model neuron k seperti pada Gambar 2.4. Pada gambar ini pengaruh dari threshold digambarkan sebagai: (1) menambah sinyal input baru dengan nilai –1, dan (2) menambah bobot sinapsis baru yang nilainya sama dengan nilai threshold θ_k.



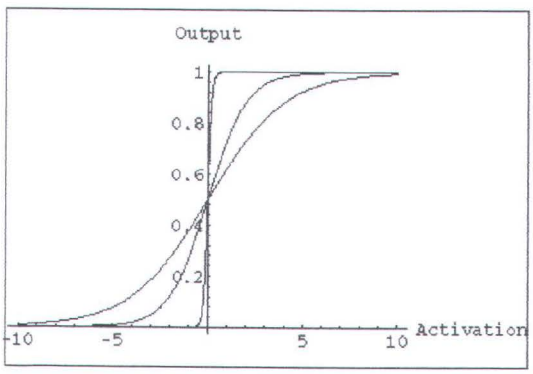
Gambar 2.3 Model neuron dengan threshold

2.3 FUNGSI AKTIFASI

Fungsi aktivasi $\varphi(.)$, menyatakan output neuron dengan level aktivasi sebagai inputannya. Fungsi sigmoid merupakan salah satu tipe fungsi aktivasi. Fungsi ini seringkali dipakai dalam penyelesaian problem yang terdapat pada sistem jaringan syaraf tiruan. Salah satu contoh fungsi sigmoid adalah

$$\varphi(\upsilon) = \frac{1}{1+\exp(-a\upsilon)} \tag{2.8}$$

dimana a merupakan parameter *slope* dari fungsi sigmoid. Dengan mengubah nilai a, akan diperoleh fungsi sigmoid dengan slope yang berbeda-beda, seperti yang ditunjukkan pada Gambar 2.5. Sebagai batasan saat nilai parameter slope mendekati tak terhingga, fungsi sigmoid akan berubah menjadi fungsi threshold. Seperti yang telah dijelaskan bahwa fungsi threshold mempunyai nilai 0 atau 1, maka fungsi sigmoid mempunyai nilai yang kontinyu dari 0 sampai 1.



Gambar 2.1 Fungsi Sigmoid

Fungsi aktivasi pada persamaan (2.8) mempunyai rentang output dari 0 sampai +1. Nilai output ini tidak hanya berkisar dari 0 namun juga bisa dirubah untuk mempunyai nilai rentang dari –1 sesuai kebutuhan dengan menerapkan persamaan berikut

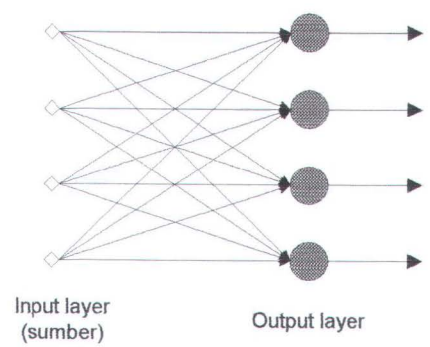
$$\varphi(\upsilon) = \tanh\left(\frac{\upsilon}{2}\right) = \frac{1-\exp(-\upsilon)}{1+\exp(-\upsilon)} \tag{2.9}$$

2.4 ARSITEKTUR JARINGAN SYARAF TIRUAN

Jaringan syaraf tiruan minimal tersusun atas layer input dan layer output. Selain kedua layer tersebut terdapat layer yang dinamakan layer *hidden* (tersembunyi). Biasanya layer hidden ini menghubungkan antara layer input dengan layer output.

2.4.1 Jaringan Single Layer Feedforward

Struktur neuron dari jaringan syaraf berkaitan erat dengan algoritma pembelajaran yang digunakan untuk melatih jaringan. Sehingga dapat dikatakan algoritma pembelajaran yang digunakan terbentuk secara struktural. Sebuah layer pada jaringan syaraf merupakan kumpulan neuron yang tersusun dalam bentuk lapisan. Bentuk sederhana dari layer ini adalah terdapat layer input yang mendapat masukkan dari sumber kemudian diteruskan ke output layer dari neuron, namun tidak dapat mengalami proses kebalikan yaitu dari output layer diteruskan ke input layer. Jaringan seperti ini dinamakan jaringan *feedforward.* Pada Gambar 2.6 digambarkan struktur jaringan yang mempunyai empat buah noda pada tiap layer input dan layer output.

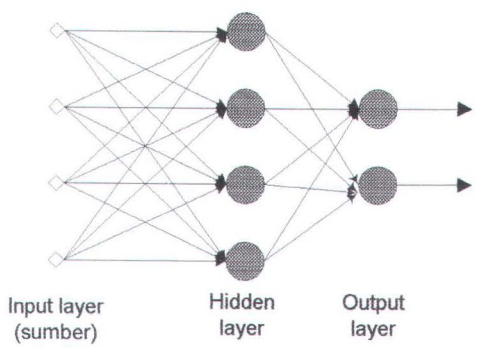


Gambar 2.1 Single layer neuron

Jaringan semacam ini dinamakan jaringan *single layer*. Pada jaringan semacam ini tidak terdapat proses komputasi karena tiap nilai output layer mengacu pada layer input yang bersangkutan. Memori asosiasi linear merupakan salah satu contoh dari jaringan syaraf tiruan single layer. Pada sistem ini, jaringan mengasosiasikan pola output (vektor) dengan pola input (vektor), dan informasi disimpan di dalam jaringan dengan melakukan perubahan pada bobot sinapsis dari jaringan tersebut.

2.4.2 Jaringan Multilayer Feedforward

Perbedaan model multilayer (Gambar 2.7) dengan single layer adalah adanya layer yang tersembunyi (*hidden* layer), dimana komputasi yang berada pada layer ini dinamakan hidden neuron atau unit hidden. Fungsi dari hidden neuron ini adalah untuk mengolah input external dari input layer yang kemudian output dari hidden layer ini menjadi input bagi output layer. Dengan menambah beberapa hidden layer, jaringan tersebut dapat mengolah data orde tinggi, kemampuan ini diperlukan apabila ukuran layer input sangat besar.



Gambar 2.1 Jaringan Multilayer

2.5 METODA PELATIHAN

Metoda pelatihan jaringan syaraf tiruan didefinisikan sebagai proses modifikasi nilai bobot dan bias dari jaringan. Metoda pelatihan ini dilakukan untuk melatih jaringan dengan spesifikasi tujuan tertentu.

2.5.1 Pelatihan Supervised

Istilah dalam jaringan syaraf dinamakan *supervised learning.* Yang fungsinya belajar (*learned*) dari contoh-contoh sebelumnya. Kumpulan contoh *training set*) terdiri dari pasangan variabel idependen (input) dan variabel dependen (output). Metoda pelatihan digunakan untuk menyesuaikan nilai bobot dan bias supaya nilai output dari jaringan syaraf mendekati nilai output target. Sebagai contoh, variabel independen dalam fungsi relasi

$$y = f(\mathbf{x}) \tag{2.10}$$

adalah **x** (vektor) dan variable dependennya adalah y (skalar). Nilai dari variabel y tergantung dari fungsi f untuk setiap komponen dari variabel vektor

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}$$

Himpunan training set T yang terdiri dari p pasangan x dan y (diindeks dengan i dihitung dari 1 sampai p), dinyatakan dengan

$$T = \{(x_i, \acute{y}_i)\}_{i=1}^{p} \tag{2.11}$$

Alasan diberikan tanda diatas y adalah nilai output dari training set diperkirakan umumnya disertai gangguan (*noise*). Dengan kata lain, nilai

sesungguhnya untuk input **x** yaitu y tidak diketahui. Training set hanya memberikan nilai ý yang nilainya sama dengan y ditambah noise.

Misalkan terdapat p input $x_1, x_2, \ldots, x_p$ dimana tiap input ini mempunyai bobot $w_1, w_2, \ldots, w_p$. Tiap sinyal input yang memiliki bobot ini kemudian dijumlah untuk menghasilkan sinyal output y. Hasil akhir yang ingin dicapai dari sistem ini adalah menentukan nilai optimal dari bobot $w_1, w_2, \ldots, w_p$ sehingga perbedaan antara output sistem y dengan respon yang diinginkan d dapat diminimalkan.

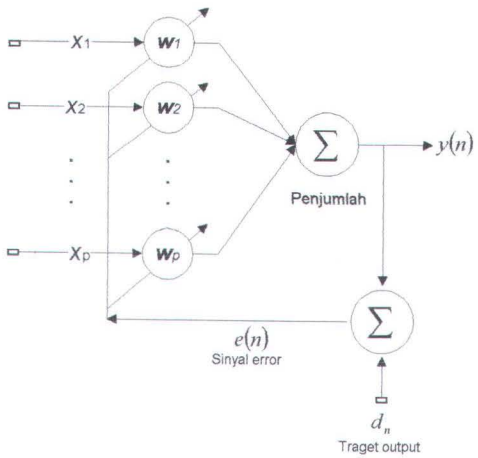
Tahap pelatihan ini menggunakan algoritma *Least Mean Square*[LMS-97] yaitu

$$\bar{w}_k(n+1) = \bar{w}_k(n) + \eta[d(n) - y(n)]x_k(n), \qquad k = 1, 2, \ldots, p \qquad (2.12)$$

Dimana η merupakan konstanta laju pelatihan dan $y(n)$ merupakan output pada saat perhitungan n iterasi yaitu

$$y(n) = \sum_{k=1}^{p} \bar{w}_k(n)x_k(n) \qquad (2.13)$$

Untuk diketahui penggantian $w_{k(n)}$ dengan $\bar{w}_{k(n)}$ untuk menyatakan bahwa persamaan (2.12) menggunakan estimasi nilai bobot dari sistem. Gambar 2.8 menunjukkan lingkungan operasi dari algoritma LMS seperti yang dijabarkan pada persamaan (2.12) dan (2.13).



Gambar 2.1 Adaptif spasial filter

Algoritma LMS dinyatakan sebagai berikut:

1. Inisialisasi

$$\overline{w}_k(1) = 0 \quad \text{..... untuk k = 1, 2, ..., p}$$

2. Filtering

Selama n = 1, 2, ...

a. $y(n) = \sum_{j=1}^{p} \overline{w}_j(n) x_j(n)$

b. $e(n) = d(n) - y(n)$

c. $\overline{w}_k(n+1) = w_k(n) + \eta e(n) x_k(n)$

sampai didapat hasil yang konvergen dengan $e(n) = \eta$

2.5.2 Pelatihan Unsupervised

Pada pelatihan *unsupervised* atau disebut juga *self-supervised* tidak terdapat umpan balik dalam proses pelatihannya. Dengan kata lain, tidak terdapat contoh

khusus dari fungsi yang hendak dipelajari. Untuk menerapkan metoda pelatihan ini digunakan pembelajaran kompetitif yang pada umumnya terjadi pengelompokan data berdasarkan kemiripan antara data yang satu dengan data yang lain. Sebagai contoh, digunakan jaringan syaraf tiruan yang terdiri dari dua layer yang dinamakan layer input dan layer kompetisi. Layer input menerima data yang ada. Layer kompetisi berisi neuron yang saling berkompetisi dengan neuron yang lain dalam merespon masukan yang diberikan oleh layer input. Dengan menggunakan strategi '*winner takes all*' maka neuron yang berhasil memenangkan kompetisi tersebut akan di set aktif dan neuron yang lain akan di set pasif. Sehingga pada akhirnya akan diperoleh himpunan cluster yang beranggotakan nilai masukkan yang dimenangkannya. Salah satu algoritma pengelompokan data pelatihan adalah *K-Mean clustering* [SUB-99]. Algoritma *K-Mean clustering* dinyatakan sebagai berikut:

1. Memilih sembarang k pusat kluster $M_1(1)$, $M_2(1)$, ..., $M_k(1)$. K vektor ini bisa dipilih secara random dari pola vektor yang ada, atau dipilih berdasarkan informasi yang terdapat pada tiap-tiap kluster. Index iterasi m di-set 1.

2. Untuk semua vektor X, vektor x_j dikelompokkan ke kluster k jika

$$\left|x_j M_k(m)\right| < \left|x_j - M_i(m)\right| \qquad (3.1)$$

untuk semua $i = 1, 2, \ldots, k$ *dan* $i \neq k$

3. Mengubah vektor mean kluster ke-i dengan persamaan

$$M_k(m+1) = \frac{1}{N_k} \sum_{k=0}^{n-1} x_k \qquad (3.2)$$

dengan n merupakan jumlah anggota untuk tiap kluster.

4. Ulangi langkah-langah diatas sampai diperoleh hasil yang konvergen.

2.6 REGRESI NONPARAMETRIK [LES-98]

Terdapat dua pecahan divisi dari problem regresi di statistika: paramatrik dan nonparametrik. Pada regresi parametrik bentuk fungsi relasional antara variabel dependen dan independen diketahui namun mungkin terdapat parameter yang nilainya tidak diketahui dan dapat diperkirakan dari training set. Umumnya pada problem parametrik, parameter bebas, maupun variabel dependen dan independen, memiliki interpretasi yang penting.

Yang membedakan regresi nonparametrik adalah tidak ada (atau sangat kecil) informasi mengenai bentuk dari fungsi sesungguhnya yang sedang dikalkulasi. Fungsi tersebut dibentuk dengan menggunakan persamaan yang mengandung parameter bebas dengan cara sedemikian sehingga memenuhi kelas fungsi yang mana model fungsi tersebut dapat dinyatakan dalam arti yang luas. Umumnya melibatkan banyak parameter bebas yang tidak mempunyai arti khusus dalam hubungannya dengan problem tersebut. Pada regresi parametrik terdapat sejumlah kecil parameter khas dan sering parameter ini mempunyai interpretasi tersendiri.

Jaringan syaraf tiruan, termasuk jaringan syaraf tiruan fungsi basis radial, tidak termasuk model parametrik dan untuk *weight* (dan parameter lainnya) tidak mempunyai arti khusus dihubungkan dengan problem yang menerapkannya. Nilai perkiraan dari bobot jaringan syaraf tiruan (atau parameter dari banyak model nonparametrik) tidak pernah menjadi tujuan utama dalam *supervised learning.*

Tujuan utama adalah memperkirakan fungsi yang sedang dikerjakan.(atau paling tidak mencari output dari input yang diinginkan).

2.7 MODEL LINEAR [LES-98]

Model linear untuk fungsi $y(\mathbf{x})$ berbentuk

$$F(\mathbf{x}) = \sum w_j h_j(\mathbf{x}) \tag{2.12}$$

Model f dinyatakan sebagai kombinasi linear dari himpunan m fungsi jadi (seringkali dinamakan fungsi basis dengan analogi bahwa vektor tersusun atas sebuah kombinasi linear dari vektor basis).

Fleksibilitas dari f adalah dapat dipasang atau dipakai dengan fungsi yang bermacam-macam, dengan hanya memilih nilai bobot yang berbeda-beda. Jika fungsi basis dapat berubah selama proses learning maka model tersebut nonlinear.

Himpunan fungsi yang ada dapat digunakan sebagai himpunan basis, jika himpunan tersebut dapat dibedakan satu dengan yang lainnya. Statistik umum dipenuhi dengan model linear yang fungsi basisnya polinomial ($h_j(\mathbf{x}) = x^j$). kombinasi dari gelombang sinusoidal (deret Fourier),

$$h_j(x) = \sin\left(\frac{2\pi j(x-\theta_j)}{m}\right) \tag{2.13}$$

seringkali digunakan pada aplikasi pemrosesan sinyal. Contoh yang lebih sederhana, hampir merupakan polynomial paling sederhana yaitu

$$f = ax + b \tag{2.14}$$

merupakan model linear yang mempunyai dua fungsi basis yaitu

$$h_1(x) = 1,$$

$$h_2(x) = x,$$

dan bobotnya adalah $w_1 = b$ dan $w_2 = a$. Hal ini, tentu saja, merupakan model sangat sederhana dan kurang cukup fleksibel untuk digunakan dalam *supervised learning.*

Model linear lebih mudah dipelajari secara matematis. Dalam kasus tertentu, jika *supervised learning* diselesaikan dengan kwadrat terkecil maka dimungkinkan untuk menurunkan dan menyelesaikan kumpulan persamaan untuk nilai bobot optimal yang dinyatakan secara tidak langsung dengan himpunan data pelatihan.

2.8 FUNGSI RADIAL [LES-98]

Fungsi radial adalah kelas fungsi yang spesial. Mempunyai karakteristik yaitu menghasilkan nilai yang bertambah atau berkurang secara monoton seiring dengan jaraknya dari titik pusat. Pusat, skala jarak, dan bentuk yang pasti dari fungsi radial merupakan parameter dari model tersebut, semuanya telah ditentukan jika modelnya linear.

Fungsi gaussian (Gambar 2.9) merupakan fungsi radial yang umum, dengan menggunakan input skalar, yaitu

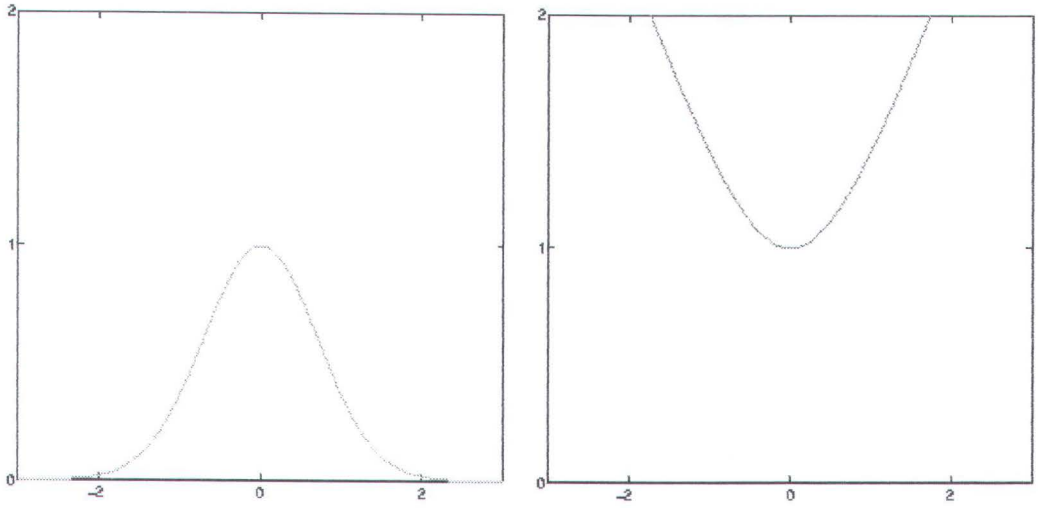
$$h(x) = \exp\left(-\frac{(x-c)^2}{r^2}\right) \tag{2.15}$$

Parameternya adalah pusat c dan radius r. Gambar 2.9.a menjelaskan fungsi radial gaussian dengan pusat $c = 0$ dan radius $r = 1$.

Sebuah fungsi radial gaussian secara monoton berkurang nilai fungsinya menurut jaraknya dari. pusat. Lain halnya, fungsi radial multiquadric, dengan skalar input:

$$h(x) = \frac{\sqrt{r^2 + (x-c)^2}}{r} \tag{2.16}$$

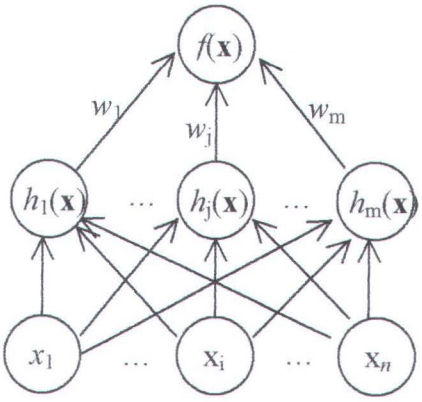
$h(\mathbf{x})$ = secara monoton bertambah nilainya sesuai jaraknya dari pusat. Fungsi radial gaussian bersifat lokal (hanya memberikan respon tertentu disekitar pusat) dan lebih umum digunakan dibanding dengan fungsi radial multiquadric yang mempunyai respon global. Gaussian lebih masuk akal secara biologis karena mempunyai respon yang terbatas.



Gambar 2.1 Fungsi radial gaussian (kiri) dan multiquadric (kanan)

2.9 JARINGAN SYARAF TIRUAN FUNGSI BASIS RADIAL

Fungsi radial merupakan kelas fungsi yang sederhana. Pada prinsipnya, dapat diterapkan pada berbagai macam model (linear/nonlinear) dan berbagai macam jaringan syaraf tiruan (single/multi layer). Namun jaringan syaraf tiruan Fungsi Basis Radial (selanjutnya disingkat dengan JST-FBR) secara tradisionil telah diasosiasikan dengan fungsi radial pada jaringan single layer seperti yang ditunjukkan pada Gambar 2.10



Gambar 2.1 Tradisional JST-FBR

Setiap n komponen dari input vektor **x** menuju m fungsi basis dimana outputnya secara linear digabung dengan bobot $\{w_j\}_{j=1}$ dan ditujukan ke output jaringan $f(\mathbf{x})$. Sebuah JST-FBR dikatakan nonlinear jika fungsi basisnya dapat bergerak atau berubah ukurannya atau jika terdapat lebih dari satu hidden layer.

BAB III

PENERAPAN JST-FBR UNTUK PENGENALAN WAJAH

BAB III

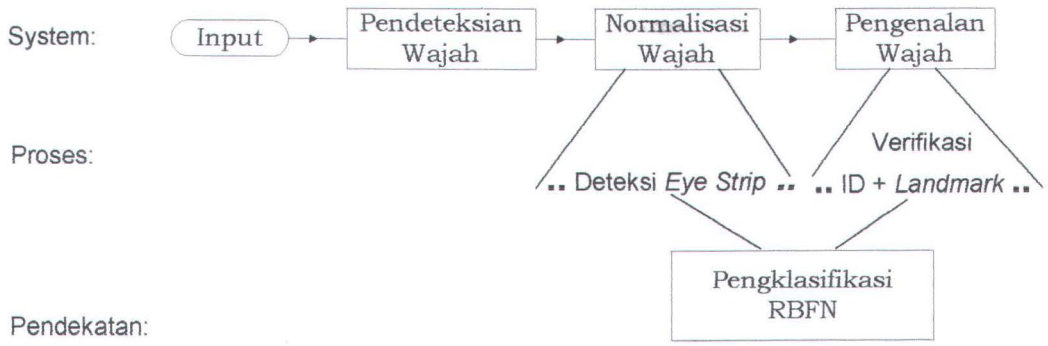
PENERAPAN JST-FBR UNTUK PENGENALAN WAJAH

Mengenali seseorang tampaknya dilakukan secara langsung – manusia melakukannya setiap saat di dalam dunia bisnis maupun dalam lingkungan sosialnya. Pendeteksian yang terotomatisasi, bagaimanapun juga, membutuhkan sistem komputer untuk memeriksa kumpulan karakteristik tersimpan yang besar dan mengambil salah satu yang paling cocok dengan karakteristik wajah yang sedang dideteksi. Proses pengenalan wajah manusia [SRI-97] dapat dibagi menjadi dua bagian, yaitu:

(1) Pencocokan (*match*): Citra dari wajah seseorang diambil (probe) dan pengidentifikasiannya dilakukan dengan mencari citra yang sama dari kumpulan citra yang telah tersimpan (galeri). Ukuran dari galeri dan jumlah probe umumnya berukuran 50-100, namun pada banyak aplikasi dimungkinkan untuk mencapai lebih dari 1000 besarnya.

(2) Pengamatan (*surveillance*): dibandingkan dengan proses pengidentifikasian wajah seseorang, sistem ini telah melibatkan vertifikasi dan memeriksa apakah probe yang sedang diproses termasuk kedalam galeri yang lebih kecil, kadang kala diberi label atau dikelompokkan ke dalam kumpulan galeri baru (*Intruder).* Probe ini berkisar dari wajah individual sampai ke wajah terkenali yang mempunyai karakteristik tertentu. Sistem pengamatan ini umumnya

dipenuhi dengan kumpulan basis data citra wajah yang besar dan kebanyakan dari wajah-wajah tersebut jika dianalisa memberikan nilai yang negatif.



Gambar 3.1 Sistem pengenalan wajah

Sistem pengenalan wajah (Gambar 3.1) biasanya dimulai dengan mendeteksi sebuah citra dengan mencari yang mana yang merupakan wajah lalu menandainya dengan sebuah kotak, kemudian dilanjutkan dengan menormalisasi citra wajah berdasarkan geometri dan perubahan pencahayaan berdasarkan kotak yang menandai wajah/lokasi mata, dan terakhir mengidentifikasi wajah tersebut dengan menggunakan representasi citra yang tepat dan algoritma klasifikasi. Pada tugas akhir ini hanya membahas rancangan perangkat lunak yang dikembangkan untuk mewujudkan dan mengimplementasikan proses pengenalan wajah pada tingkat dimana pekerjaan pengklasifikasian diperlukan.

3.1 ALASAN PENGGUNAAN JARINGAN SYARAF TIRUAN FUNGSI BASIS RADIAL UNTUK PENGENALAN WAJAH

Sistem pengenalan wajah ini menerapkan salah satu model jaringan syaraf buatan yaitu JST-FBR dengan pertimbangan beberapa kelebihan yang dimilikinya yaitu:

- Kemampuannya untuk mengelompokkan citra-citra serupa sebelum mengklasifikasikannya dan kemungkinan pengembangan selanjutnya dimana wajah manusia dikelompokkan secara iteratif ke dalam jenis kelamin, ras, dan umur, sebelum dilakukan tahap akhir dari pengenalan wajah. [JON-96]
- Komputasi yang sederhana. Pengklasifikasi FBR mempunyai arsitektur yang sangat mirip dengan jaringan tiga-lapis propagasi balik, namun pada proses pelatihan *supervised* hanya melibatkan satu layer saja. Sehingga dibutuhkan waktu yang lebih singkat untuk mencapai nilai yang konvergen. [JON-96]
- JST-FBR dapat memproses data-set tanpa dilakukan preproses terlebih dahulu, dengan hanya berdasarkan pada nilai pixel.

3.2 FITUR WAJAH

Masukkan bagi sistem pengenalan buatan ini berupa citra *grayscale* yang tiap-tiap piksel didalamnya mempunyai intensitas antara 0-255. Dari berbagai macam ukuran citra baiknya dilakukan pemilihan atau pembatasan ukuran citra karena tidak semua bagian didalamnya akan diproses seluruhnya. Kriteria dalam pemilihan ini adalah bagian dari wajah yang membedakan atau yang menjadi ciri tiap-tiap individu.

Termasuk didalam kriteria tersebut adalah daerah mata, hidung dan mulut karena ketiga ciri tersebut diasumsikan sudah dapat membedakan tiap-tiap individu Dari kenyataan yang ada manusia memiliki ekspresi wajah yang bermacam-macam, sehingga terdapat variasi pada daerah mata dan terutama

daerah mulut. Mata misalnya memiliki kombinasi antara kelopak mata terbuka dan tertutup. Pada daerah mulut lebih banyak lagi variasi yang tejadi. Namun dari berbagai variasi ekspresi yang ada diharapkan sistem pengenalan buatan ini dapat mengatasi permasalahan di atas.

Citra wajah yang layak diproses oleh sistem pengenalan buatan ini adalah citra wajah dimana ketiga kriteria tersebut terdapat didalamnya. Dengan pertimbangan dari data yang didapat maka citra wajah yang diproses (citra normalisasi) memiliki ukuran 62x68 pixel dengan fitur wajah yaitu mata, hidung dan mulut terdapat didalamnya.

3.3 PROSES PENGENALAN WAJAH

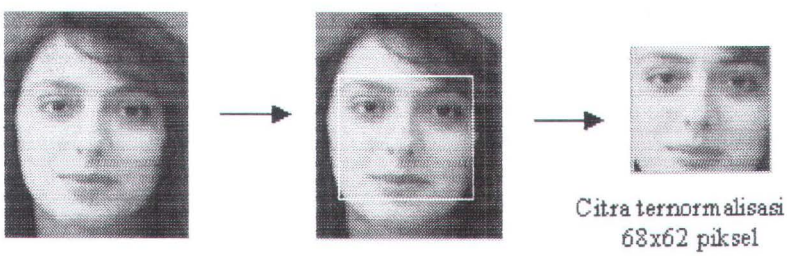
Proses pengenalan wajah dengan menggunakan arsitektur jaringan syaraf tiruan, menerima masukkan berupa citra wajah dua dimensi dengan posisi wajah menghadap ke depan. Berbeda dengan proses pengenalan sesungguhnya yang dilakukan oleh manusia, yang memperoleh inputan berupa bidang tiga dimensi, terdapat penurunan informasi (tekstur wajah misalnya) yang didapat oleh sistem pengenalan buatan karena adanya transformasi bentuk dari bidang tiga dimensi menjadi dua dimensi.

Untuk mengatasi hilangnya sebagan besar informasi akibat transformasi dimensi maka diperlukan variasi data untuk diolah oleh sistem pengenalan buatan. Variasi ini meliputi intensitas dan arah pencahayaan, ekspresi wajah yang berbeda-beda dan posisi dari wajah itu sendiri seperti menoleh ke kiri, ke kanan, tengadah, tunduk dan lain-lain. Jika dari semua variasi yang ada hendak diolah oleh sistem pengenalan buatan, maka diperlukan sebuah sistem yang sangat

kompleks. Untuk itu perlu adanya semacam seleksi data dari sekian banyak variasi atau membatasi macam bentuk variasi untuk mendapatkan suatu sistem pengenalan buatan yang akurat. Salah satu contoh pembatasan variasi adalah jarak antara objek dengan kamera pengambil gambar untuk semua data haruslah sama.

3.3.1 Ekstraksi Fitur

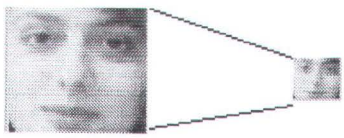
Dari blok citra ternormalisasi dengan ukuran 68x62 yang berisi daerah mata, hidung dan mulut, dilakukan proses ekstraksi fitur yang akan digunakan sebagai data pelatihan. Hasil ekstraksi ini kemudian disimpan ke dalam basis data Microsoft Access. Untuk mendapatkan citra ternormalisasi ini harus dilakukan secara manual dengan menempatkan kotak pencari ciri (Gambar 3.2) sedemikian cara sehingga area didalam kotak pencari ciri mencakup lokasi mata, hidung dan mulut yang posisi ketiga daerah ciri tersebut tersusun secara proporsional.



Gambar 3.1 Normalisasi citra

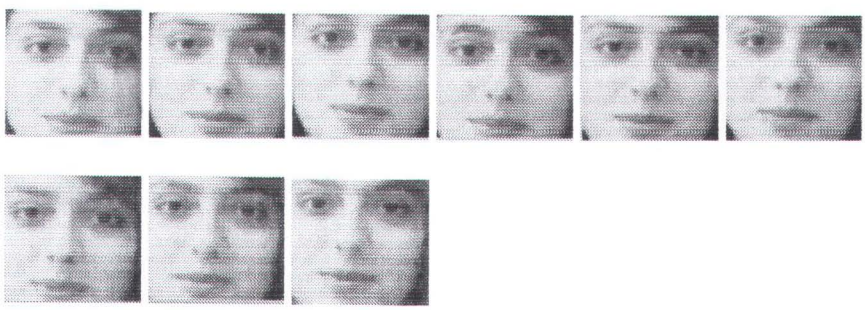
Setelah didapatkan citra normalisasi selanjutkan dilakukan pre-proses [JON-96] (Gambar 3.3) yaitu *downsampling* citra normalisasi yang semula berukuran 68x62 menjadi berukuran 20x20. Meskipun JST-FBR sendiri dapat mengolah data tanpa pre-proses terlebih dahulu, namun hal ini layak dilakukan untuk menghemat waktu komputasi. Dari citra normalisasi yang berukuran 68x68 jika tanpa pre-

proses maka akan menghasilkan vektor satu dimensi sebesar 4624, jika dilakukan pre-proses maka besarnya vektor hanya sebesar 400.



Gambar 3.2 Proses Downsampling

Karena pada proses normalisasi wajah dilakukan secara manual maka terdapat variasi baru yang bersifat eksternal dengan pertimbangan apabila proses normalisasi wajah dilakukan untuk yang kedua kalinya dengan data yang sama maka akan didapatkan citra normalisasi yang berbeda. Untuk mengatasi masalah ini dilakukan pemvariasian citra ternormalisasi yang diperoleh menjadi 9 buah citra ternormalisasi. (Gambar 3.4). Dengan perlakuan seperti ini tiap citra mempunyai sembilan macam variasi. Variasi tersebut diperoleh dengan menggeser kotak pencari dari posisi normalnya yaitu: kiri atas, kiri, kiri bawah, atas, normal, bawah, kanan atas, kanan dan kanan bawah.



Gambar 3.3 Variasi eksternal

3.3.2 Proses Pelatihan

Sebelum sebuah sistem pengenalan wajah berbasis jaringan syaraf buatan dipakai dalam proses pengenalan, terlebih dahulu dikenai proses pelatihan dengan tujuan supaya sistem menyimpan parameter-parameter yang terbentuk saat proses pelatihan. Sehingga saat proses pengenalan dilakukan sitem pengenalan wajah akan membandingkan hasil pengenalan dengan parameter-parameter yang tersimpan sebelumnya.

3.3.2.1 Pelatihan Unsupervised

Dari basis data wajah yang telah diperoleh saat ekstrasi fitur sebelumnya, dipilih mana yang akan dijadikan data pelatihan bagi sistem pengenal wajah. Hasil pilihan ini kemudian disimpan dalam bentuk file teks (*.trn), yang berisi vektor linear berikut id dari pemiliknya. File training ini yang kemudian dibaca saat proses pelatihan dilakukan. Selanjutnya pada tahap pertama proses pelatihan, digunakan metoda unsupervised dengan menerapkan algoritma *k-mean clustering.* Tujuan dari pelatihan ini adalah untuk mengelompokkan citra-citra serupa dari sekian banyak citra yang yang akan dikenai proses pelatihan pada sistem pengenalan wajah JST-FBR.

Pertama-tama dilakukan inisialisasi terhadap data pelatihan, yang berupa vektor linear berdimensi 400, dengan melakukan proses normalisasi terlebih dahulu. Kemudian tiap cluster yang terdapat pada layer hidden, sejumlah banyaknya data pelatihan, nilai meannya diset sesuai dengan nilai salah satu data pelatihan yang terasosiasi dengannya. Misalnya terdapat 5 data pelatihan yang masing-masing memiliki 9 macam variasi eksternal maka akan terdapat 5 cluster

pada layer hidden yang mempunyai relasi satu-satu, dengan nilai mean yang di set sama dengan nilai salah satu dari 9 macam variasi eksternalnya. Setelah inisialisasi selesasi maka dilakukan k-means clustering bedasarkan jarak eucledian antar data pelatihan yang hanya mewakili satu individu saja. Setelah proses k-mean clustering selesai tiap-tiap cluster pada layer hidden mempunyai nilai mean sebanyak dimensi dari vektor linear yang diproses. Pengelompokkan kluster pada algoritma k-mean hanya berpengaruh pada variasi data dari individu yang sama dengan kata lain tidak mungkin citra orang *A* dikelompokkan kedalam citra orang *B*. Pada proses pelatihan ini algoritma k-mean diiterasi sebanyak jumlah data pelatihan.

3.3.2.2 Pelatihan Supervised

JST-FBR mempunyai arsitektur yang sangat mirip dengan jaringan tiga-lapis propagasi balik. Noda pada layer hidden, dinamakan BF noda/*cluster*, yang menghasilkan respon lokal terhadap masukan dengan menggunakan kernel Gaussian. Setiap unit hidden dapat dianggap sebagai *receptive field* (RF) lokal. Fungsi basis yang digunakan adalah Gaussian, dimana tingkat aktivasi h_i dari unit hidden i diberikan oleh:

$$h_i = -\exp\left[\sum_{k=1}^{D} \frac{(x_k - \mu_{ik})^2}{2g\sigma_{ik}^2}\right] \tag{3.3}$$

Dimana g adalah konstanta pembanding selisih (*variance*), x_k merupakan komponen ke 'k' dari vektor input X=[x_1, x_2, ...,x_D], sedangkan μ_{ik} adalah komponen ke 'k' dari mean (k-mean). σ_{ik}^2 merupakan varian pada unit hidden i yang didapat dari persamaan[LES-98]:

$$\sigma_{ik}{}^2 = \frac{1}{n-1}\sum_{k=1}^{v}(x_{ik} - \mu_i)^2 \qquad (3.4)$$

n merupakan jumlah anggota yang dimiliki oleh unit hidden i (didapatkan saat pelatihan unsupervised), sedangkan v merupakan ukuran vektor linier yang besarnya 400. Dari kedua rumus di atas dapat dilihat bahwa variabel μ_{ik} menyatakan pusat dan variabel σ_{ik} menyatakan jari-jari.

Pada proses pelatihan untuk memperoleh nilai bobot optimum w_1, w_2, ..., w_m, layer output hanya diiterasi sebanyak jumlah node pada layer hidden, karena nilai output dari tiap unit hidden pada tahap pelatihan hanya bernilai 0 atau 1 dengan pemahaman: unit hidden bernilai satu berasosiasi dengan citra yang terdapat dalam data pelatihan, sedangkan unit hidden bernilai nol tidak mempunyai asosiasi dengan citra yang bersangkutan. Apabila salah satu unit hidden bernilai satu, maka unit hidden yang lain akan bernilai 0. Asumsi yang dipakai adalah tiap satu unit hidden menyimpan ciri dari satu citra wajah. Dengan asumsi seperti ini maka pemilihan data pelatihan memerankan bagian yang sangat penting karena akan mempengaruhi hasil dari proses pengenalan. Variasi data pelatihan sangatlah berpengaruh pada jumlah unit hidden.

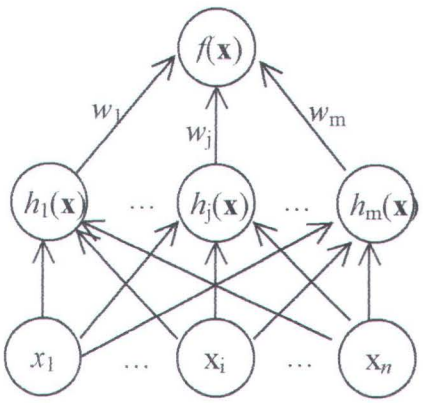
Dengan menggunakan fungsi aktifasi *binary* sigmoid untuk memperoleh nilai keluaran dimana

$$y(\upsilon) = \frac{1}{1+\exp(-\upsilon)} \qquad (3.5)$$

dan

$$\upsilon = \sum_{k=1}^{p} w_k h_k \qquad (3.6)$$

maka pada layer output nilai yang dihasilkan mempunyai rentang antara 0 – 1. Dengan memperhatikan arsitektur JST-FBR pada Gambar 3.5 maka pencarian nilai bobot optimum w_1, w_2, ..., w_m, dengan menggunakan algotitma LMS untuk mendapatkan minimum error dari target output, relatif sangat cepat. Target output yang dimaksudkan disini adalah nilai normalisasi dari id number tiap-tiap data pelatihan. Setelah proses selesai, nilai-nilai yang didapat disimpan kedalam sebuah file teks (*.net). File ini berisi jumlah cluster pada layer input, jumlah cluster pada layer hidden, nilai bobot, mean, variance dari masing-masing cluster pada layer hidden dan nilai bias. File ini yang nantinya akan digunakan saat proses pengenalan wajah.



Gambar 3.1 Jaringan Syaraf Tiruan Fungsi Basis Radial

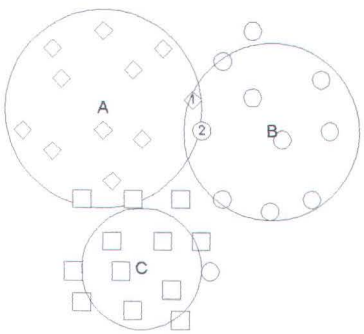
3.3.3 Proses Pengenalan

Pertama-tama parameter-parameter jaringan yang disimpan didalam file (*.net) dibaca oleh sistem. Kemudian memilih satu citra yang akan diproses dan diambil fitur wajahnya dengan menggunakan ekstraksi fitur wajah. Vektor linear yang didapat kemudian diproses oleh setiap unit hidden. Dengan menggunakan rumus (3.3) dan (3.4) setiap unit hidden akan menghasilkan nilai berkisar antara 0

– 1. Apabila dari rumus diatas salah satu unit hidden menghasilkan nilai lebih besar dari *level aktivasi hidden* maka nilainya di-set 1 sedangkan unit hidden yang lain di-set 0. Jika tak ada unit hidden yang mempunyai nilai lebih besar dari level aktivasi hidden maka semua hidden di-set 0 atau dengan kata lain citra wajah tersebut tidak dikenali oleh sistem.

Level aktivasi hidden adalah level terkecil dimana nilai hasil perhitungan dari sebuah unit hidden dianggap sudah memenuhi syarat sebagai penanda bahwa input yang diproses terasosiasi dengannya. Besarnya nilai level aktivasi hidden sangat mempengaruhi hasil yang didapat.

Level aktivasi hidden dapat dikatakan sebagai pembeda ciri antar pola yang satu dengan yang lain. Pengelompokan pola pada JST-FBR didasarkan pada pusat dan radius. Yang dimaksud pusat adalah pola-pola itu sendiri, sedangkan radius menyatakan banyaknya ciri yang dimiliki oleh pola tersebut. Maka dengan radius yang semakin besar terdapat kemungkinan ciri pada radius tersebut dimiliki oleh pola yang lain. Seperti yang terlihat pada Gambar 3.6, semakin besar radius maka pada salah satu ciri, yang terdapat pada dua lingkaran yang saling beririsan, dapat dikelompokkan ke dalam dua kelas pola yaitu kelas A dan kelas B.



Gambar 3.1 Distribusi Pola

Untuk menghindari hal ini maka diusahakan radius antara tiap-tiap pola tidak saling beririsan, seperti pada pola C, namun dengan pertimbangan tidak semua ciri yang dimiliki oleh pola tersebut tercakup di dalamnya.

Setelah masing-masing cluster memproses fungsi basisnya, fungsi linear dari rumus (3.6) dihitung yang kemudian dilanjutkan dengan perhitungan fungsi aktivasi yang terdapat pada rumus (3.5). Nilai yang diperoleh dari sistem pengenalan wajah JST-FBR ini merupakan id dari citra yang diproses. Apabila didalam basis data tidak ditemukan id tersebut maka citra yang diproses tidak dapat dikenali oleh sistem.

BAB IV

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK

BAB IV

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK

Pengembangan perangkat lunak pada Tugas Akhir ini merupakan implementasi teori–teori tentang pengenalan wajah dengan menggunakan JST-FBR yang telah diuraikan pada bab sebelumnya. Pembahasan dibatasi pada rancangan perangkat lunak yang dikembangkan untuk mewujudkan dan mengimplementasikan proses pengenalan wajah pada tingkat dimana pekerjaan pengklasifikasian diperlukan. Pada bab ini akan dibahas tentang perancangan sistem perangkat lunak, dan pembuatan sistem perangkat lunak dengan menjelaskan tahapan–tahapan proses dan struktur data.

4.1 DESKRIPSI SISTEM

Perancangan dan pembuatan perangkat lunak pengenalan wajah menggunakan spesifikasi perangkat lunak sebagai berikut :

- Processor Intel Celeron 400 MMX
- RAM 128 MB
- Hardisk 6.4 GB

Spesifikasi lain yang juga digunakan dalam pembuatan perangkat lunak ini adalah :

- Sistem operasi Windows 98

- Bahasa pemrograman Borland C++ Builder 4.0

4.2 PERANCANGAN PERANGKAT LUNAK

Isi dari sub-bagian ini berisi perancangan perangkat lunak sistem pengenalan wajah yang meliputi struktur data yang digunakan dan aliran prosesnya.

4.2.1 Perancangan Data

Struktur data terpenting dalam sistem pengenalan wajah ini terdiri dari struktur data JST-FBR dan struktur data pola pelatihan. Struktur data JST-FBR digunakan untuk merepresentasikan arsitektur dari JST-FBR, sedangkan struktur data pola pelatihan digunakan untuk merepresentasikan pola-pola pelatihan yang digunakan selama proses pelatihan. Arsitektur JST-FBR dan pola-pola pelatihan disimpan kedalam bentuk file text dengan format tertentu yang akan diterangkan pada pembahasan masing-masing struktur data. Berikut ini adalah penjelasan kedua struktur data tersebut

4.2.1.1 Struktur Data JST-FBR

JST-FBR merupakan jaringan syaraf tiruan terstruktur yang terdiri atas sejumlah cluster seperti yang telah diterangkan pada bab-bab sebelumnya. Informasi yang terdapat di dalamnya dapat dibagi menjadi tiga bagian yaitu informasi topologi FBR, unit FBR dan unit proses yang terdapat di dalam JST-FBR.

4.2.1.1.1 Kelas Unit RBF

Data yang terdapat di dalam kelas ini memberikan informasi beberapa parameter yang terdapat antara layer input dengan layer hidden yaitu mengenai jumlah noda pada layer input, nilai input yang merupakan isi dari vektor linear, nilai varian (radius), nilai output yang digunakan untuk menyimpan nilai keluaran sistem yang diharapkan dan, nilai weight vektor yang merupakan titik pusat. Dalam perangkat lunak ini, jumlah input didapat dari ukuran vektor linear yaitu sejumlah 400 input data-data tersebut adalah:

1) Number of input, berisi jumlah cluster pada layer input ata besarnya vektor linear. Bentuk data : integer.
2) Input value, merupakan nilai dari vektor linear yang didapat dari ekstraksi fitur wajah yang hendak dilakukan proses pelatihan atapun proses pengenalan. Bentuk data : array float.;
3) Mean, nilai mean atau pusat pada layer input. Bentuk data : array float.
4) Variance, nilai simpangan data. Bentuk data : array float.
5) Output value, nilai yang dihasilkan oleh masing-masing cluster pada layer. hidden. Bentuk data : array float.

4.2.1.1.2 Kelas Unit Proses

Kelas ini menyimpan parameter yang terdapat di antara layer hidden dengan layer output. Data-data yang terdapat di dalamnya tersusun sebagai berikut:

a) Jumlah input unit, jumlah cluster pada layer hidden yang menjadi input unit bagi layer output. Bentuk data : Integer

b) Unit proses input, nilai output yang dihasilkan oleh tiap-tiap cluster pada layer hidden. Bentuk data : array float.

c) Bobot, nilai bobot pada layer hidden. Bentuk data : array float.

d) Bias. Bentuk data : float

e) Koreksi bobot, nilai koreksi bobot pada waktu pelatihan jaringan. Bentuk data array float.

f) Koreksi bias, nilai koreksi bias pada saat pelatihan jaringan. Bentuk data : float.

g) Jumlah perkalian bobot, menyimpan hasil penjumlahan dari perkalian bobot dengan fungsi basisnya yaitu fungsi radial. Bentuk data : float.

h) Sinyal output, nilai yang dihasilkan oleh layer output. Bentuk data : float

i) Error info, menyimpan hasil selisih kesalahan yang terjadi saat proses pelatihan untuk kemudian dijadikan perhitungan saat mengkoreksi nilai bobot dan bias. Bentuk data : float.

4.2.1.1.3 Kelas Topology FBR

Kelas ini menyusun keseluruhan struktur dari sistem JST-FBR, mulai dari layer input sampai dengan layer output dengan menggunakan dua kelas sebelumnya .

a) Sinyal dimensi. Jumlah node pada layer input. Bentuk data : integer.

b) Jumlah cluster layer hidden. Bentuk data integer.

c) Rata-rata jumlah error kuadrat. Kuadrat selisih nilai output yang dibandingkan dengan nilai target. Bentuk data : float

d) Cluster hidden. Bentuk data : array unit FBR.

e) Cluster output layer. Bentuk data array : Topology FBR.

Setiap cluster yang terdapat pada layer hidden digunakan untuk mengelompokkan data berdasarkan kedekatan masing-masing data. Setiap cluster direpresentasikan oleh pusat cluster*(mean)* dan varian(radius). Mean dan varian merupakan parameter statistik yang merepresentasikan tingkat penyebaran data dalam cluster tersebut. Banyaknya cluster yang terdapat pada FBR dideklarasikan di klas topology FBR yaitu jumlah maksimum cluster. Dimensi sinyal yang terdapat didalamnya menyatakan jumlah vektor input ditambah vektor output.

Nilai keluaran terdiri dari dua jenis yaitu keluaran yang dihasilkan oleh layer hidden dan keluaran yang dihasilkan oleh sistem jaringan syaraf tiruan pengenal wajah. Keluaran dari layer hidden dihasilkan dari perhitungan fungsi basis masing-masing cluster hidden dari masukkan data pelatihan. Keluaran dari masing-masing cluster ini akan dibandingkan nilainya dengan keluaran cluster yang lain dalam proses pengenalan wajah untuk mendapatkan cluster yang memberikan hasil keluaran positif yang terbesar. Keseluruhan cluster yang membentuk arsitektur jaringan FBR yang terbentuk setelah mengalami proses pelatihan disimpan kedalam file text yang disebut file jaringan yang mempunyai tipe *.net, dengan format seperti di bawah ini :

Header		
Identitas	dimensi sinyal	Jumlah cluster

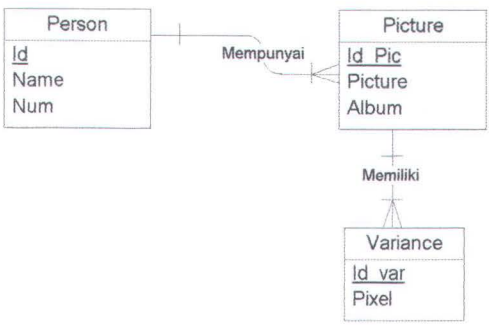
Informasi cluster										
Bobot layer hidden ke-0	...	Bobot layer hidden ke-n	Varian layer hidden ke-0	...	Varian layer hidden ke-n	Mean layer hidden ke-0	...	Mean layer hidden ke-n	Nilai output maksimal	Nilai output minimal

Gambar 4.1 Format file jaringan

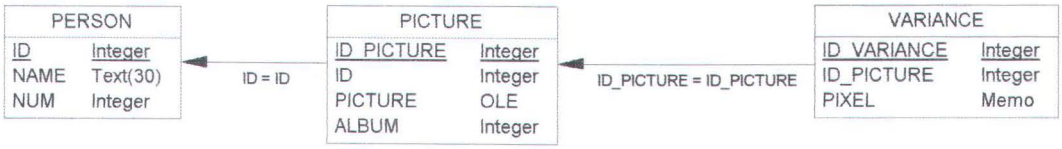
Keterangan masing-masing field di atas adalah sebagai berikut. Bagian header file jaringan berisi tiga field yaitu: (1) field identitas yang berisi bilangan penanda yaitu 1, (2) field dimensi sinyal yang menyatakan jumlah vektor linear ditambah dengan jumlah output yaitu 401, (3) field jumlah cluster yang menyatakan jumlah cluster yang terdapat didalam jaringan ini. Bagian informasi cluster berisi informasi yang dimiliki cluster-cluster yang ada di dalam jaringan. Masing-masing cluster berisi: bobot cluster, varian cluster pusat cluster(mean), nilai terbesar yang dihasilkan oleh jaringan, dan nilai terkecil yang dihasilkan oleh jaringan. Fungsi dari kedua nilai output ini adalah sebagai parameter normalisasi yang dilakukan saat proses pelatihan.

4.2.1.2 Struktur Data Pola Pelatihan

Pola pelatihan didapat dari pemilihan sejumlah citra wajah yang disimpan dalam bentuk basis data. Basis data ini berisi citra wajah berikut variasi eksternal nya dan deskripsi nama pemilik wajah tersebut disertai dengan ekstraksi fitur wajahnya yang berupa vektor linear. ER diagram dari basis data tersebut dapat dilihat pada Gambar 4.2. Pola-pola pelatihan merupakan vektor linear. Pola-pola ini dipilih dari basis data wajah berdasarkan nama pemilik wajah.



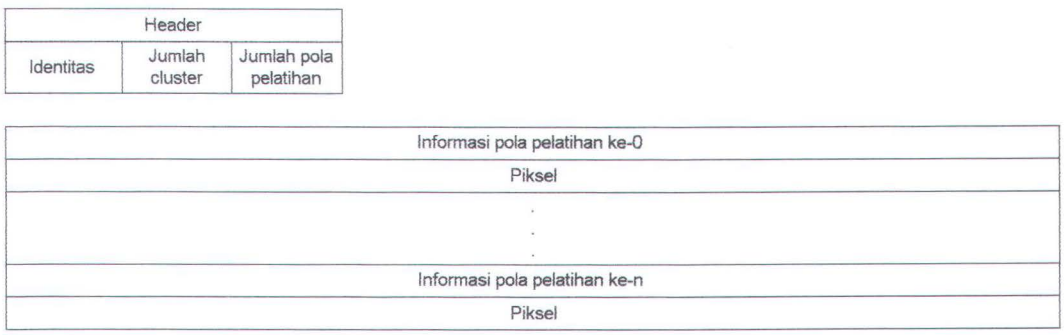
Gambar 4.1 ER diagram basis data wajah



Gambar 4.2 PDM dari ER diagram basis data wajah

Data yang akan diolah pada proses pelatihan ini berisi vektor linear hasil extraksi fitur wajah dari beberapa wajah yang sudah dipilih yang akan dikenali. Hasil pemilihan tersebut kemudian disimpan kedalam betuk file text yang bertipe *.trn. yang merupakan file data pelatihan. Saat proses pelatihan jaringan syaraf tiruan pengenalan wajah data pelatihan ini dibaca dari file penyimpan data pelatihan dan kemudian disimpan ke dalam memori. Deklarasi struktur data ini berupa *array* berdimensi satu dengan jumlah elemen sesuai dengan pola citra tersebut.

Struktur file penyimpanan pola tersebut digambarkan sebagai berikut:



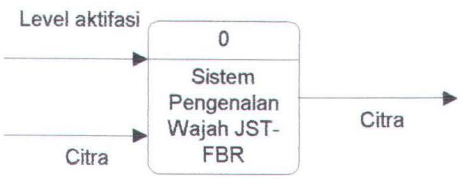
Gambar 4.3 Format file pola pelatihan

Keterangan tiap-tiap field di atas adalah sebagai berikut. Bagian header file pola terdiri dari field identitas yang berisi bilangan penanda file pelatihan yaitu 0. Field jumlah cluster yang menyatakan jumlah cluster pada layer hidden. Field jumlah pola pelatihan yang menunjukkan jumlah pola data pelatihan yang terdapat dalam

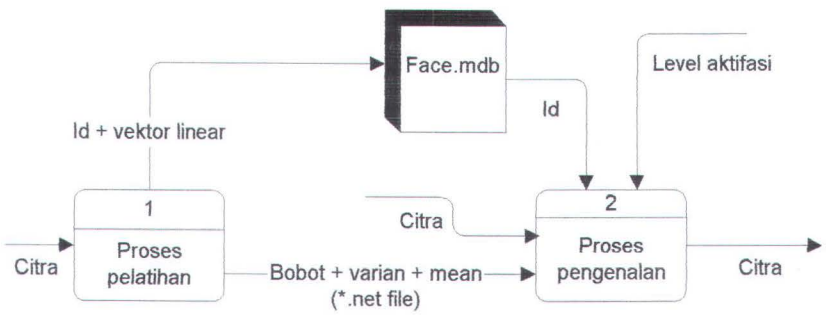
file. Setelah bagian header file kemudian mengikuti berikutnya adalah vektor-vektor linear dari fitur pola-pola yang disimpan.

4.2.2 Perancangan Proses

Pengenalan wajah ini dapat dibagi menjadi dua bagian proses yaitu proses pelatihan jaringan dan proses pengenalan wajah. Dengan pertimbangan bahwa sebelum proses pengenalan wajah dilakukan, harus dilakukan proses pelatihan jaringan untuk mendapatkan bentuk jaringan yang diinginkan untuk pengenalan wajah. Namun proses pelatihan ini tidak selalu harus dilakukan apabila sudah terbentuk jaringan dari proses pelatihan sebelumnya. Berikut (gb 4.6 – gb 4.7) merupakan data alir diagram dari JST-FBR :



Gambar 4.1 DAD level 0



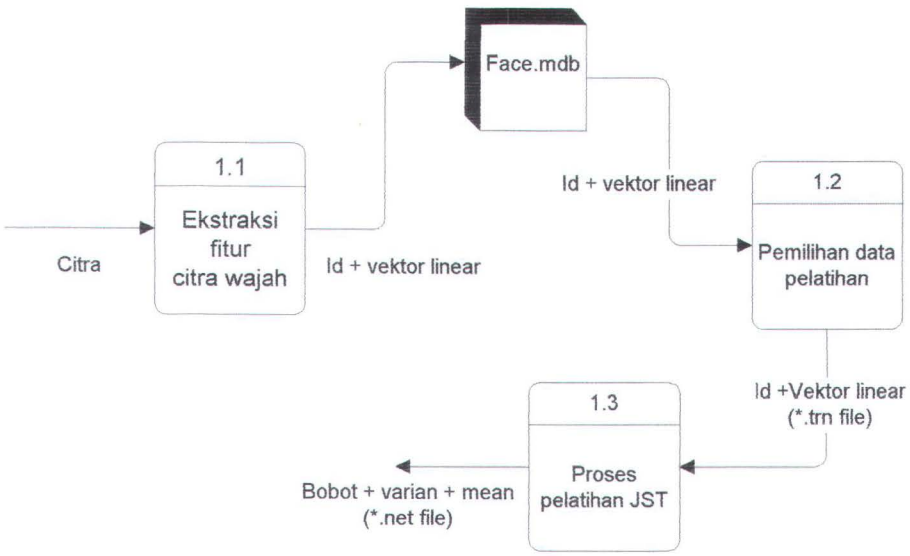
Gambar 4.2 DAD level 1 Sistem pengenalan wajah JST-FBR

Proses pengenalan digunakan untuk mengenali citra wajah yang dimasukkan ke dalam proses ini dengan menggunakan jaringan syaraf tiruan yang sudah dilatih, sedangkan proses pelatihan jaringan digunakan untuk membentuk jaringan syaraf tiruan pengenalan wajah dengan menggunakan jaringan fungsi

basis radial berdasarkan data-data pelatihan yang berupa fitur citra yang hendak dikenali.

4.2.2.1 Proses Pelatihan Jaringan

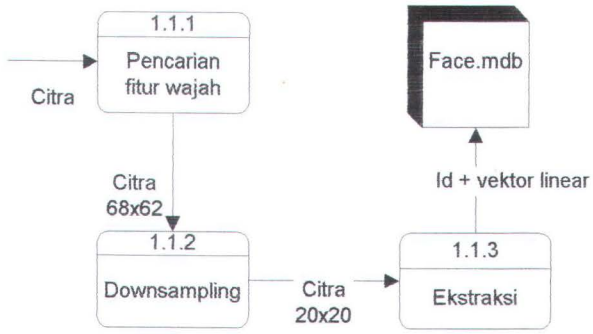
Proses pelatihan jaringan pengenalan wajah meliputi ekstraksi fitur dan jaringan pengenalan wajah. Data yang dimasukkan untuk ekstraksi fitur adalah data fitur wajah yang diperoleh secara manual dengan mengambil area bujursangkar yang berisi dua mata, hidung dan mulut dari setiap citra wajah yang ada. Sedangkan data untuk pelatihan jaringan pengenalan wajah adalah fitur wajah yang didapat dari proses ekstraksi fitur yang berupa vektor satu linear. Proses pelatihan jaringan tersebut diperlihatkan pada Gambar 4.7 berikut ini.



Gambar 4.1 DAD level 2 Proses pelatihan

Proses ekstraksi fitur wajah dilakukan dengan menentukan area-area yang akan diekstraksi pada setiap citra wajah yang tersedia. Area yang ditandai meliputi area wajah yang berisi mata kanan dan kiri, hidung dan mulut untuk

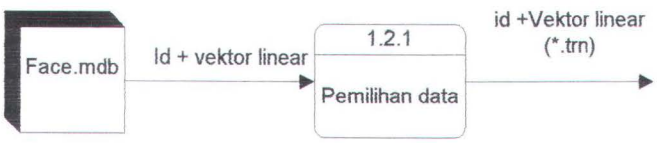
proses normalisasi yang akan menghasilkan citra wajah ternormalisasi. Dari citra wajah ternormalisasi tersebut dapat diekstraksi fitur wajah (Gambar 4.8) yang



Gambar 4.2 DAD Level 3 Ekstraksi fitur citra wajah

ternormalisasi dengan melakukan proses down-sampling terlebih dahulu yaitu dari ukuran asal 68x68 piksel menjadi 20x20 piksel. Fitur ini disimpan dalam bentuk vektor linear yang berdimensi 400.

Jaringan pengenal wajah FBR terdiri atas beberapa cluster hidden yang diasosiasikan dengan banyaknya data pelatihan. Sehingga banyak sedikitnya cluster hidden tergantung dari kualitas pemilihan data pelatihan. Sedapatnya data yang dipilih untuk dijadikan data pelatihan adalah data yang dapat mewakili data-data yang lain yang dianggap serupa. Dengan kata lain tiap-tiap cluster hidden yang terdapat pada jaringan mewakili salah satu ciri fitur wajah. Berikut merupakan proses pemilihan data pelatihan :



Gambar 4.3 DAD level 3 Pemilihan data pelatihan

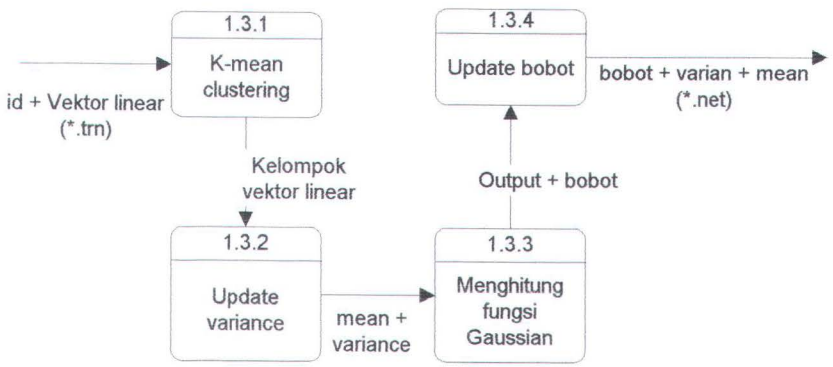
Dalam proses pelatihan jaringan pengenal wajah terdapat dua macam pelatihan yaitu: pelatihan bebas *(unsupervised)* dan pelatihan terawasi (supervised).

4.2.2.1.1 Pelatihan Unsupervised

Proses pelatihan (gambar 4.10) *unsupervised* ini digunakan untuk mendapatkan parameter jaringan pengenal wajah, yaitu :

- μ sebagai nilai pusat dan
- σ yang merupakan nilai radius dari μ itu sendiri.

Proses ini terjadi pada hubungan antara input layer dengan hidden layer . Namun terlebih dahulu dilakukan proses k-mean *clustering* yang akan melakukan proses



Gambar 4.1 DAD level 3 Proses pelatihan JST

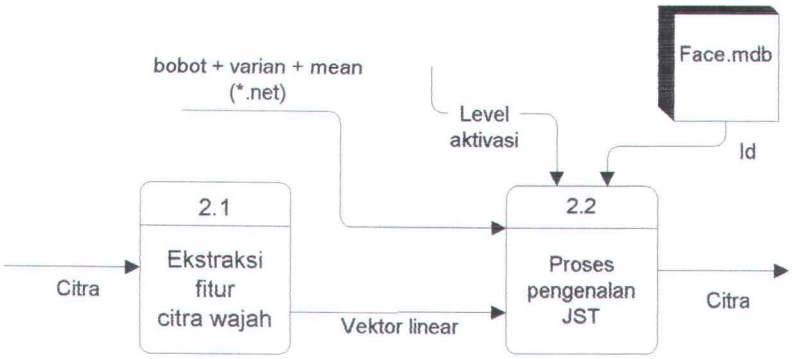
pengelompokkan berdasarkan kedekatan atau kemiripan antar masing-masing data pelatihan yang berisi fitur wajah ternormalisasi. Kemiripan yang dimaksud didapat dari jarak *eucledian* (eucledian distance), semakin kecil jaraknya semakin mirip data tersebut. Pengelompokkan dibatasi pada tiap individu yang diwakili oleh citranya. Dengan kata lain tidak mungkin orang A dikelompokkan kedalam anggota orang B. Data yang memiliki nilai kemiripan tinggi dijadikan satu kelompok. Setelah proses k-mean selesai dapat dilanjutkan dengan mencari nilai μ dan σ kemudian dilanjutkan ke proses pelatihan *supervised.*

4.2.2.1.2 Pelatihan Supervised

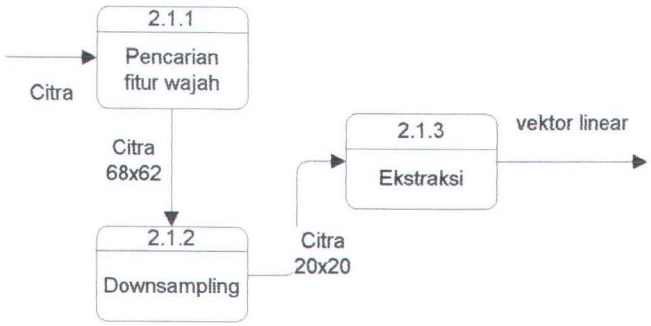
Proses ini berjalan pada layer hidden dengan layer output. Fungsi dari pelatihan ini adalah untuk mendapatkan parameter jaringan pengenal wajah yaitu ω yang merupakan nilai bobot pada layer hidden. Pada proses pelatihan ini jaringan hanya dilatih berdasarkan pada nilai keluaran dari layer hidden yang 'hanya' bernilai 0 atau 1 dengan ketentuan hanya satu dari sekian cluster hidden yang nilainya berbeda, sehingga pada proses pelatihan *unsupervised* ini relatif berlangsung sangat singkat.

4.2.2.2 Proses Pengenalan Wajah

Proses pengenalan wajah dilakukan terhadap citra wajah yang dimasukkan oleh pemakai pada sistem ini. Keluaran sub-sistem ini berbentuk hasil klasifikasi terhadap citra masukkan yang ditampilkan ke media tampilan. Diagram alir data secara garis besar ditunjukkan dengan Gambar 4.11 dan untuk detail pada tahap pengenalan ditunjukkan pada Gambar 4.12 dan Gambar 4.13. Proses pengenalan wajah dimulai dengan memasukkan parameter jaringan pengenal wajah terlebih dahulu yaitu : (1) pusat, (2) radius dan (3) nilai bobot. Kemudian dilakukan proses ekstraksi fitur wajah terhadap citra masukan yang akan menghasilkan citra wajah yang ternormalisasi yang akan diolah oleh proses pengenalan wajah.

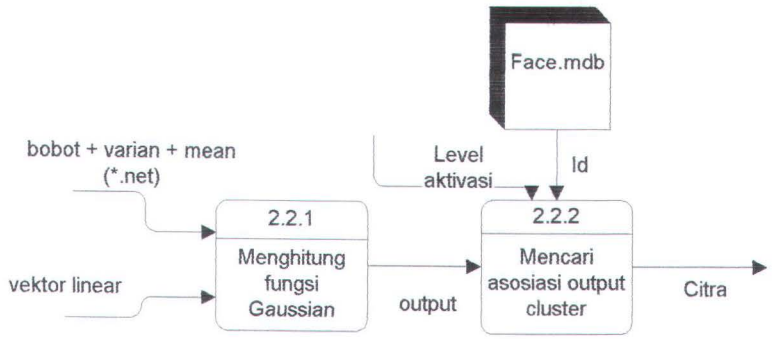


Gambar 4.1 DAD level 2 Proses pengenalan



Gambar 4.2 DAD level 3 Ekstraksi fitur wajah

Fitur wajah yang dihasilkan berbentuk citra dengan ukuran 68x68 piksel. Kemudian fitur wajah ternormalisasi ini direduksi menjadi vektor linear yang berdimensi 400. Fitur vektor ini kemudian diproses oleh setiap cluster yang terdapat pada layer hidden. Hasil pemrosesan setiap cluster digunakan sebagai parameter hasil klasifikasi. Jika terdapat cluster ke-i yang menghasilkan nilai terbesar maka citra dikenali sebagai orang yang mempunyai ciri yang dimiliki oleh cluster ke-i tersebut. Jika nilai yang dihasilkan tidak memenuhi target nilai aktivasi maka citra wajah yang dimasukkan tidak berhasil dikenali oleh jaringan.



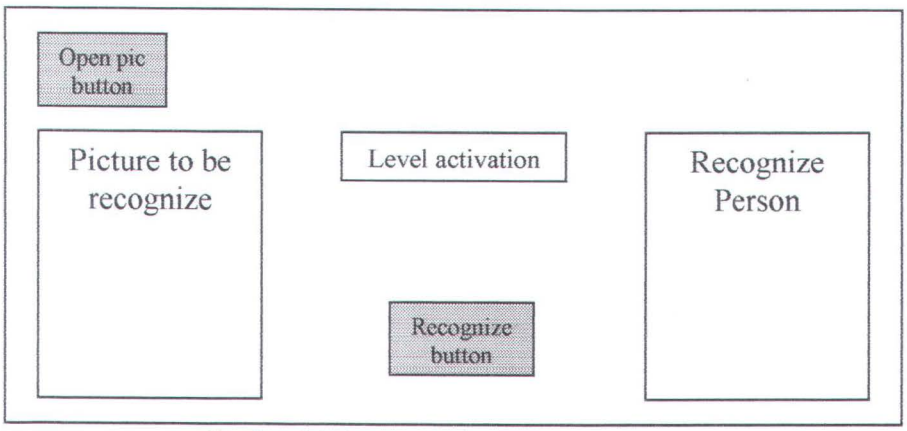
Gambar 4.3 DAD level 3 Proses pengenalan JST

4.2.3 Perancangan Antar Muka

Perancangan antar muka mempunyai tujuan untuk mempermudah penggunaan sistem yang dibangun dari implementasi rancangan perangkat lunak. Rancangan menu yang terdapat dalam perangkat lunak ini terdiri dari :

- Menu Utama

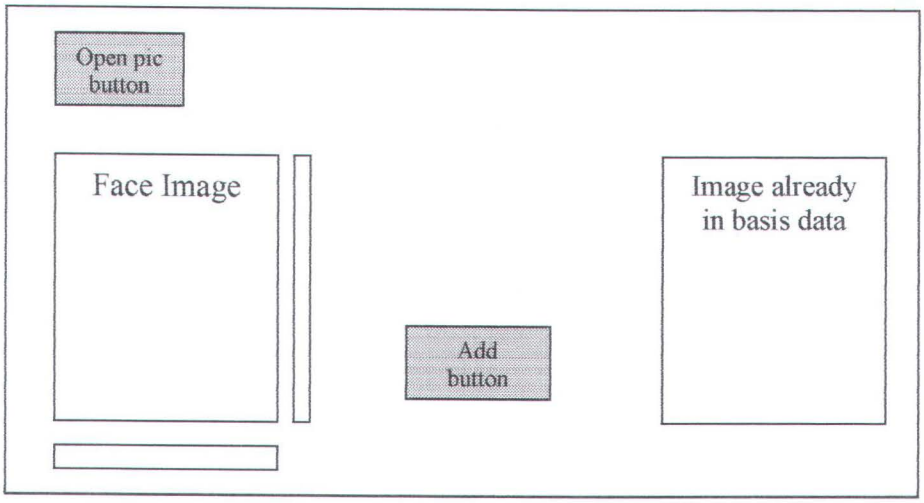
  Menu ini mucul saat perangkat lunak dijalankan pertama kali yang digunakan untuk melakukan proses pengenalan wajah.



Gambar 4.1 Rancangan layout menu utama

- Menu Pengumpulan Data

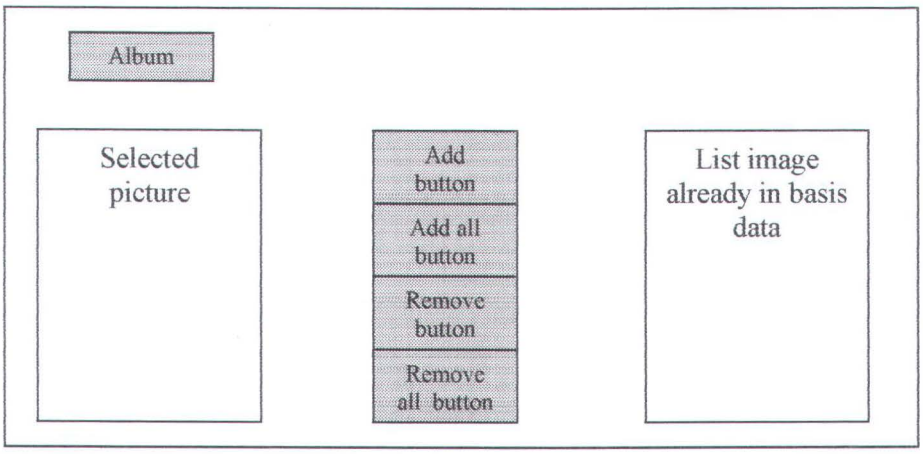
  Di dalam menu ini proses pengumpulan data pelatihan dilakukan. Proses yang dilakukan adalah memilih dari sejumlah citra wajah yang ada kemudian data citra terpilih tersebut dimasukkan kedalam basis data wajah.



Gambar 4.2 Rancangan layout menu pengumpulan data

- Menu Pemilihan Data Pelatihan

  Data pelatihan yang berisi dari vektor-vektor linear dari beberapa citra wajah dilakukan pada menu ini dengan memilih sejumlah citra wajah yang terdapat dalam basis data wajah.



Gambar 4.3 Rancangan layout pemilihan data pelatihan

4.3 PEMBUATAN PERANGKAT LUNAK

Sub bab ini membahas lebih lanjut tentang struktur data yang digunakan dan fungsi dasar yang diaplikasikan dalam implementasi rancangan perangkat lunak ke dalam bahasa pemrograman.

4.3.1 Implementasi Data

Penggunaan kelas–kelas dalam implementasi data yang akan digunakan dalam tugas akhir ini mempunyai tujuan agar pemrograman lebih terstruktur, mempermudah perhitungan dan penyimpanan data.

4.3.1.1 Struktur Data Jaringan FBR

Struktur data arsitektur JST-FBR direpresentasikan dengan kelas dalam bahasa borland C++ sebagai berikut:

1. Kelas processing_unit.

   Variabel yang terdapat di dalam kelas ini berhubungan dengan perhitungan nilai keluaran pada output layer.

Implementasi	Keterangan
class Processing_units public:	
int number_of_input_units	- Jumlah input node (cluster) pada output layer
float *processing_unit_input	- nilai output yang dihasilkan node hidden layer
float *weight_of_inputs	- nilai bobot pada hidden layer
float *weight_correction_term	- Koreksi bobot
float bias_correction_term	- Koreksi bias
float sum_of_weighted_inputs	- jumlah nilai perkalian output dan bobot pada hidden layer
float bias	- nilai bias
float output_signal	- nilai yang dihasilkan output layer
float error_information_term	

2. Kelas radial_basis_unit

Kelas ini menyimpan nilai parameter-parameter yang terdapat pada hidden layer

Implementasi	Keterangan
class Radial_Basis_units public:	
float *input_value	- berisi data masukkan pada input layer
float *Mean	- nilai mean pada input layer
float *variance	- nilai simpangan data
float output_value	- jumlah nilai dari fungsi basis radial pada hidden layer
int number_of_inputs	- jumlah node pada input layer

3. Kelas Radial_Basis_Topology

Kelas ini merupakan topologi dari fungsi basis radial dimulai dari implementasi data pada input layer sampai dengan output layer.

Implementasi	Keterangan
Class Radial_Basis_Topology: public Processing_units public:	
int dimensions_of_signal	- jumlah node pada input layer
int maximum_number_of_clusters	- jumlah node pada hidden layer
float error_difference_squared	- kuadrat selisih nilai output dengan nilai target
Radial_Basis_units *node_in_cluster_layer	
Processing_units node_in_output_layer	

4.3.1.2 Implementasi Dari Deklarasi Kelas

Sub-bab ini merupakan penjelasan dari fungsi dan prosedur yang terdapat dalam tiap-tiap kelas terpenting yang menyusun sistem jaringan syaraf tiruan pengenalan wajah.

1. Kelas Data_type

   Kelas ini berurusan dengan data input yang akan dimasukkan kedalam sistem jaringan syaraf tiruan pengenalan wajah.

Implementasi	Keterangan

class Data_type
{
public:
char *filename;
int signal_dimensions;
int sample_number;
float max_output_value;
float min_output_value;
sample_data *number_of_samples;
virtual void load_data_into_array(void);
virtual void determine_sample_number(void);
virtual void normalize_data_in_array(void);
};
	
	- Besarnya vektor linear
	- jumlah sample data dalam data pelatihan
	- nilai max yang dihasilkan oleh jaringan
	- nilai min yang dihasilkan oleh jaringan
	- data pelatihan jaringan
	- mengisi array dengan data pelatihan
	- mnghitung jumlah data dalam data pelatihan
	- normalisasi data dalam data pelatihan

2. Kelas Sample_data

   Kelas ini menyimpan data vektor linear dan id kelompok (digunakan saat proses k-mean)

```
class sample_data
{
  public:
  float *data_in_sample;
  int   cluster;
  ~sample_data();
};
```

3. Kelas Signal_data

Kelas ini digunakan untuk memberikan nilai inisialisasi bobot input dan bobot output dari tiap-tiap cluster dari layer hidden termasuk bias.

```
class signal_data signals
{
 public:
   float bedlam(long *idum);
   int gaset;
   signal_data();
};
```

4. Kelas Trainning

Kelas ini berhubungan atau digunakan pada saat proses pelatihan. Fungsinya sebagai variabel data input pelatihan bagi proses pelatihan jaringan syaraf tiruan pengenal wajah.

Implementasi	Keterangan

class Training : public Data_type, signal_data
{
public:
void request_training_data(char *file);
float minimum_average_squared_error;
float rate_of_learning;
};
	
	- load data pelatihan
	- nilai kuadrat selisih error terkecil
	- konstanta laju pelatihan

5. Kelas Testing

Seperti kelas trainning kelas ini digunakan sebagai variabel data input saat proses pengenalan wajah dilakukan.

```
class Testing : public Training
{
 public:
 char *resultsname;
 void request_testing_data(void);
 float average_squared_error;
};
```

6. Kelas Radial_basis_topology

Topologi dari JST-FBR diimplementasikan oleh kelas berikut ini. Kelas Radial_basis_topologi ini mempunyai turunan kelas yaitu kelas Processing_units dan Radial_Basis_unit.

Implementasi	Keterangan

class Radial_Basis_Topology: public Processing_units
{
public:
int cluster_champ;
int dimensions_of_signal;
int maximum_number_of_clusters;
float error_difference_squared;
float MaxOutput, MinOutput;
Radial_Basis_units *node_in_cluster_layer;
Processing_units node_in_output_layer;
void transfer_Gaussian_to_Output_layer(float percent, int test,int cluster);
void upload_network(char *getname);
void savenet(char *file);
void calculate_output_error_information_term(float target_value);
~Radial_Basis_Topology();
};
	
	- unit hidden yang berasosiasi
	- jumlah node input layer
	- jumlah node pada hidden layer
	- kuadrat selisih error
	- nilai max/min output jaringan
	- unit hidden
	-unit proses pada output layer
	- menghitung fungsi gaussian
	
	- load file net ke dalam jaringan
	- menyimpan parameter jaringan
	- perhitungan error yang terjadi pada output

7. Kelas Processing_unit

Kelas ini digunakan saat dilakukan proses trainning untuk memperbaharui nilai bobot dan bias. Kelas processing_unit berisi variabel yang diperlukan untuk memperbaharui nilai bobot dan bias.

Implementasi	Keterangan

class Processing_units : public signal_data
{
public:
int number_of_input_units;
float *processing_unit_input;
float *weight_of_inputs;
float *weight_correction_term;
float bias_correction_term;
float sum_of_weighted_inputs;
float bias;
float output_signal;
float error_information_term;
float calculate_output_signal_derivative();
	
	- jumlah hidden node
	- nilai output unit hidden
	- nilai bobot pada node hidden
	- nilai koreksi bobot
	- nilai koreksi bias
	- jumlah bobot*output hidden
	
	- nilai output jaringan
	- nilai error untuk perhitungan bobot
	- menghitung output jaringan derivatif

void calculate_weight_and_bias_correction_terms (float learning_rate);	- menghitung koreksi bobot & bias
void establish_array_of_processing_unit_inputs (void);	- inisialisasi array pada output layer
void establish_weight_vector_for_processing_units (void);	- inisialisasi nilai bobot
void calculate_output_signal(int test);	- menghitung nilai output jaringan
Processing_units();	
~Processing_units();	

8. Kelas Radial_basis_units

Kelas ini berisi parameter atau variabel yang dimiliki oleh tiap-tiap cluster yang terdapat pada layer hidden, juga terdapat prosedur untuk menghitung keluaran dari masing-masing cluster.

Implementasi	Keterangan
class Radial_Basis_units: public signal_data	
{	
public:	
int number_of_member;	- jumlah data pelatihan yang berasosiasi dengan unit hidden
int number_of_inputs;	- jumlah node pada input layer
float output;	- nilai output unit hidden
float output_value;	- nilai sementara fungsi gaussian
float Gaussian_transfer_output;	- nilai fungsi basis radial
float *input_value;	- vektor linear
float *Mean;	
float *variance;	
void activation(void);	- perhitungan fungsi gaussian
void calculate_sum_square_Euclidean_distance (void);	- menghitung jarak eucledian data pelatihan
Radial_Basis_units();	
~Radial_Basis_units();	
};	

9. Kelas Neural

Kelas neural ini merupakan inti dari sistem pengenalan wajah. Didalamnya terdapat proses pelatihan dan proses pengenalan wajah.

Implementasi	Keterangan
Class Neural	
{	
private:	
Training RTrain;	- variabel training

Testing RTests;
int do_k_means(int dolock);
void init_k_means();
void count_variance(void);
void train_RBF_neural_network(float percent);
void test_neural_network
(char *file, float percent);
public:
Radial_Basis_Topology RBF_Design;
void establish_Radial_Basis_network(int cluster);
};
	- variabel test
	- k-means clustering
	- inisialisasi proses k-mean
	- menghitung nilai varian
	- proses pelatihan jaringan
	- proses pengenalan jaringan
	
	- variabel RBF
	- inisialisasi jaringan

4.3.2 Implementasi Proses

Sub-bagian ini menjelaskan tentang urutan proses dalam sistem pengenalan wajah dengan menggunakan JST-FBR. Proses-proses utama dalam sistem pengenalan wajah ini dibagi menjadi dua bagian. Bagian pertama merupakan kumpulan proses pelatihan jaringan syaraf tiruan dan bagian kedua berisi kumpulan proses pengenalan wajah. Kumpulan proses pelatihan jaringan syaraf tiruan ini terdiri dari proses pengumpulan pola pelatihan dan pelatihan jaringan syaraf tiruan yang digunakan selama proses pengenalan, sedangkan kumpulan proses pengenalan wajah terdiri dari proses ekstraksi fitur wajah dan proses pengenalan wajah berdasarkan basis data wajah yang ada. Penjelasan masing-masing proses diatas dalam bentuk pseudocode akan diterangkan berikut ini.

4.3.2.1 Proses Pelatihan

- **Pengumpulan pola pelatihan**

  Pola pelatihan yang digunakan dalam perangkat lunak ini terdiri dari pola fitur wajah. Ekstraksi pola dari citra wajah yang disediakan adalah sebagai berikut: fitur wajah diambil dengan mengambil bagian dari citra wajah dengan ukuran 68x68 piksel yang berisi informasi mata, hidung dan

mulut. Dari fitur wajah yang didapat dicari varian eksternalnya. Kemudian pola tersebut direduksi sehingga ukurannya menjadi 20x20 piksel. Selanjutnya pola terreduksi tersebut diekstraksi kembali sehingga berbentuk vektor linear. Berikut pseudocode dari pengumpulan

pola pelatihan dalam bahasa C++

```
for(int x = 0; x < 3; x++)
 {
  if(x!=0)
  {
  Rn.Left += 3;
  Rn.Right += 3;
  }
  for(int y = 0; y < 3; y++)
  {
  if(y!=0)
  {
   Rn.Top += 3;
   Rn.Bottom +=3;
  }
  Image2->Canvas->CopyRect(Bingkai,DBImage1->Picture->Bitmap->
  Canvas,Rn);
  Shrink_Image();
  DataSource6->DataSet->Last();
  DataSource6->DataSet->Append();

  DBMemo1->Text="";
  for (i=0;i<20; i++)
   for (j = 0; j<20; j++)
   {
    pixel = GetRValue( Image1->Canvas->Pixels[i][j]);
    pixel = pixel ;
    DBMemo1->Lines->Add(pixel);
   }
  DBMemo1->Lines->Add(IntToStr(lock));
  DBMemo1->Lines->Add(" ");

  DBEdit11->Text = DBEdit4->Text;

  if(DBEdit12->Text == "")
   id_variance = 1;
  else
   id_variance = StrToInt(DBEdit12->Text) + 1;

  DBEdit13->Text = id_variance;

  DataSource6->DataSet->Post();
  if(y == 2)
  {
```

```
            Rn.Top -= 6;
            Rn.Bottom -=6;
          }
        }
      }
```

Sedangkan proses downsampling dari citra berukuran 68x68 piksel menjadi 20x20 piksel pseudocodenya sebagai berikut:

```
XRatio = (float) ( (float)68 /(float)20 );
YRatio = XRatio;

XPoints = (int *)malloc(20 * sizeof(int));

for( Xc=0, X = 0; Xc < 20; X+=XRatio, Xc++ )
  XPoints[Xc] = X;

YPoints = (int *)malloc(20 * sizeof(int));

for( Yc=0, Y = 0 ; Yc < 20 ; Y+=YRatio, Yc++ )
  YPoints[Yc] = Y;

for( Yc = 0 ; Yc < 20 ; Yc++ )
  for( Xc = 0 ; Xc < 20 ; Xc++)
    Image1->Canvas->Pixels[Xc][Yc] = Image2->Canvas->Pixels[XPoints[Xc]][YPoints[Yc]];
```

➢ **Pelatihan Jaringan Pengenal Wajah**

Jaringan syaraf tiruan pengenal wajah merupakan jaringan pengklasifikasi wajah. Hasil dari klasifikasi jaringan tersebut kemudian dibandingkan dengan data yang ada di dalam basis data.

- **K-mean clustering**

Proses pengelompokkan ini hanya berlaku pada pola citra yang menghasilkan output cluster yang sama atau lebih sederhananya pengelompokkan didasarkan pada citra wajah orang yang sama. Berikut adalah pseudocode k-mean clustering:

```
int champ, count;
float temp;

for(int pattern = 0; pattern < RTrain.sample_number; pattern++)
{
 for(int knodes = 0; knodes < RBF_Design.maximum_number_of_clusters; knodes++)
        for(int dim = 0; dim < RBF_Design.dimensions_of_signal; dim++)
          RBF_Design.node_in_cluster_layer[knodes].input_value[dim] =
RTrain.number_of_samples[pattern].data_in_sample[dim];
  champ =
RBF_Design.kluster_nodes_compete_for_activation(RTrain.number_of_samples[pattern].data_
in_sample[RBF_Design.dimensions_of_signal]);
  RTrain.number_of_samples[pattern].cluster = champ;
  if(RTrain.number_of_samples[pattern].cluster != champ)
    dolock++;

}

for(int knodes = 0; knodes < RBF_Design.maximum_number_of_clusters; knodes++)
{
 for(int dim = 0; dim < RBF_Design.dimensions_of_signal; dim++)
 {
   temp = 0;
   count = 0;
   for(int pattern = 0; pattern < RTrain.sample_number; pattern++)
     if(RTrain.number_of_samples[pattern].cluster == knodes)
     {
       temp += RTrain.number_of_samples[pattern].data_in_sample[dim];
       count++;
     }
   RBF_Design.node_in_cluster_layer[knodes].number_of_member = count;
   if(count == 0)
     RBF_Design.node_in_cluster_layer[knodes].input_weight_vector[dim] = 0;
   else
     RBF_Design.node_in_cluster_layer[knodes].input_weight_vector[dim] = temp/count;
 }
}
```

- **Hitung varian**

Varian ini didapat setelah dilakukan proses k-mean. Pada rumus 3.4 nilai mean (μ) didapatkan dari rata-rata jumlah vektor linear yang termasuk dalam satu kelompok cluster. Sehingga setiap cluster yang merupakan pusat dari pola mempunyai nilai varian yang berbeda. Berikut adalah pseudocode pencarian nilai varian.

```
float sum;
int knodes;
for(knodes = 0; knodes < RBF_Design.maximum_number_of_clusters; knodes++)
  for(int dim = 0; dim < RBF_Design.dimensions_of_signal; dim++)
    {
     sum = 0;
     for(int pattern = 0; pattern < RTrain.sample_number; pattern++)
      if(RTrain.number_of_samples[pattern].cluster == knodes)
        sum += pow((RTrain.number_of_samples[pattern].data_in_sample[dim]-
        RBF_Design.node_in_cluster_layer[knodes].input_weight_vector[dim]),2.0);
     if((RBF_Design.node_in_cluster_layer[knodes].number_of_member == 0) ||
        (RBF_Design.node_in_cluster_layer[knodes].number_of_member == 1))
            RBF_Design.node_in_cluster_layer[knodes].variance[dim] = sum;
     else
        RBF_Design.node_in_cluster_layer[knodes].variance[dim] = sum /
        (RBF_Design.node_in_cluster_layer[knodes].number_of_member - 1);
    }
```

- **Update bobot**

Pada proses pelatihan ini jaringan hanya dilatih berdasarkan pada nilai keluaran dari layer hidden yang 'hanya' bernilai 0 atau 1 dengan ketentuan hanya satu dari sekian cluster hidden yang bernilai 1 sedangkan yang lain harus bernilai 0. berikut adalah pseudocode update bobot:

```
float output_error, sum_of_error, real_error_difference,
      target_minimum_average_squared_error;
int cnode;
int test = 0;

RTrain.rate_of_learning = 0.3;
target_minimum_average_squared_error = 0.02;
RTrain.minimum_average_squared_error = MAXFLOAT;

do
{
 sum_of_error = 0;
 for(cnode = 0; cnode < RBF_Design.maximum_number_of_clusters; cnode++)
  {
   RBF_Design.transfer_Gaussian_to_Output_layer(percent, test, cnode);
   RBF_Design.calculate_output_error_information_term(
   RBF_Design.node_in_cluster_layer[cnode].output);
   real_error_difference = (pow(RBF_Design.error_difference_squared, 0.5)) *
   (RTrain.max_output_value - RTrain.min_output_value);
   output_error = 0.5 * pow(real_error_difference, 2.0);

   sum_of_error += output_error / float
(RBF_Design.maximum_number_of_clusters);
```

```
  RBF_Design.node_in_output_layer.calculate_weight_and_bias_correction_terms(
  RTrain.rate_of_learning);
  }
  if(sum_of_error < RTrain.minimum_average_squared_error)
    RTrain.minimum_average_squared_error = sum_of_error;

  if(RTrain.minimum_average_squared_error <=
target_minimum_average_squared_error)
    break;

} while(true);
```

4.3.2.2 Proses Pengenalan Wajah

Sub-bab ini merupakan penjelasan dari fungsi dan prosedur yang terdapat dalam tiap-tiap kelas terpenting yang menyusun sistem jaringan syaraf tiruan pengenalan wajah.

1. Kelas Data_type

a. Prosedur ini digunakan untuk menyimpan data pola yang ada di dalam file data pelatihan ke dalam memori yang untuk selanjutnya akan diolah baik oleh proses pengenalan maupun proses pelatihan jaringan

```
void Data_type :: load_data_into_array(void)
{
  // open the file containing the data
  ifstream file_ptr;
  int i;
  float hold;
  file_ptr.open(filename, ios::in);

  // create dynamic array to hold the specified number of samples
  number_of_samples = new sample_data[sample_number];

  for(i = 0; i < sample_number; i++)
  // create a dynamic array to hold the dimensions of each signal
  {number_of_samples[i].data_in_sample = new float[signal_dimensions + 1];}

  int dimensions = signal_dimensions + 1;

  //read in data from file
  file_ptr >> hold;
  file_ptr >> hold;
```

```
  file_ptr >> hold;
  for(i = 0; i < sample_number; i++)
  {
   for(int j = 0; j < dimensions; j++)
   {file_ptr >> number_of_samples[i].data_in_sample[j];}
  }
  file_ptr.close();
}
```

b. Sebelum data pelatihan diproses oleh sistem sebelumnya data tersebut dinormalisasi terlebih dahulu sehingga nilai dari data tersebut mempunyai skala 0 sampai 1

```
void Data_type::normalize_data_in_array(void)
{
 int imax, imin, trigger;
 float min = 0, max = 255, rmax, rmin;
 int total_dimension = signal_dimensions + 1;
 int i, j;

 for(j = 0; j < total_dimension; j++)
 {
  trigger = 1;

  if(j == signal_dimensions)
  {
   max = MINFLOAT;
   for(i = 0; i < sample_number; i++)
      if(i == 0)
      {
        max = number_of_samples[i].data_in_sample[j];
        min_output_value = min;
        max_output_value = max;
      }
      else
        if(number_of_samples[i].data_in_sample[j] > max)
        {
         max = number_of_samples[i].data_in_sample[j];
         max_output_value = max;
        }
   rmax = max_output_value;
   rmin = min_output_value;
   imax = int(rmax);
   imin = int(rmin);

   if((imax == 1) && (imin == 0) && (rmax <= 1.0) && (rmin <= 0.0))
    {trigger = 0;}

   if((imax == 1) && (imin == 1) && (rmax <= 1.0) && (rmin <= 1.0))
    {trigger = 0;}
```

```
    if((imax == 0) && (imin == 0) && (rmax <= 0.0) && (rmin <= 0.0))
     {trigger = 0;}
    }
    // normalize
    if(trigger != 0)
     for(i = 0; i < sample_number; i++)
       number_of_samples[i].data_in_sample[j] = number_of_samples[i].data_in_sample[j] / max);
  }
}
```

2. Kelas Signal_data

Fungsi generator bilangan acak untuk inisialisasi bobot dan bias terdapat di dalam kelas ini.

```
float signal_data::bedlam(long *idum)
{
 int xj;
 long xk;
 static long iy=0;
 static long iv[NTAB];
 float temp;

 if(*idum <= 0 || !iy)
 {
  if(-(*idum) < 1)
  {
   *idum = 1 + *idum;
  }
  else
  {
   *idum = -(*idum);
  }
  for(xj = NTAB+7; xj >= 0; xj--)
  {
   xk = (*idum) / IQ;
   *idum = IA * (*idum - xk * IQ) - IR * xk;
   if(*idum < 0)
   {
        *idum += IM;
   }
   if(xj < NTAB)
   {
        iv[xj] = *idum;
   }
  }
   iy = iv[0];
 }

 xk = (*idum) / IQ;
 *idum = IA * (*idum - xk * IQ) - IR * xk;
 if(*idum < 0)
```

```
  {
   *idum += IM;
  }
  xj = iy / NDIV;
  iy = iv[xj];
  iv[xj] = *idum;

  if((temp=AM*iy) > RNMX)
  {
   return(RNMX);
  }
  else
  {
   return(temp);
  }
}
```

3. Kelas Radial_radial_basis_topology

a. Fungsi di bawah ini digunakan untuk menentukan pengelompokkan data pelatihan berdasarkan k-mean clustering.

```
int Radial_Basis_Topology:: assign_pattern_to_cluster(float output)
{
 float minimum_distance;
 minimum_distance = MAXFLOAT;

 for(int m = 0; m < maximum_number_of_clusters; m++)
 {
  node_in_cluster_layer[m].calculate_sum_square_Euclidean_distance();
  if((node_in_cluster_layer[m].output_value < minimum_distance) &&
  (node_in_cluster_layer[m].output == output) )
  {
    cluster_champ = m;
    minimum_distance = node_in_cluster_layer[m].output_value;
  }
 }
 return(cluster_champ);
}
```

b. Prosedur establish_Radial_Basis_topology digunakan untuk menginisialisasi nilai parameter dan variabel yang terdapat di dalam jaringan sebelum dilakukan proses pelatihan jaringan dan pengenalan pola citra wajah.

```
void Radial_Basis_Topology::establish_Radial_Basis_topology(int cluster)
```

```
{
  dimensions_of_signal = 400;
  maximum_number_of_clusters = cluster;

  node_in_cluster_layer = new Radial_Basis_units[maximum_number_of_clusters];
  for(int c = 0; c < maximum_number_of_clusters; c++)
  {
   node_in_cluster_layer[c].number_of_inputs = dimensions_of_signal;
   node_in_cluster_layer[c].establish_input_output_arrays();
   node_in_cluster_layer[c].establish_Mean_array();
  }
  node_in_output_layer.number_of_input_units =  maximum_number_of_clusters;
  node_in_output_layer.establish_array_of_processing_unit_inputs();
  node_in_output_layer.establish_weight_vector_for_processing_units();
  node_in_output_layer.bias = 1.0 - (2.0 * bedlam((long*)&gaset));
}
```

c. Di bawah ini merupakan prosedur untuk menghitung fungsi aktifasi tiap-tiap cluster hidden layer terhadap pola-pola data pelatihan atau pola yang akan dikenali yang dimasukkan kedalam jaringan syaraf tiruan pengenalan wajah manusia. Prosedur ini merupakan satu kesatuan dengan prosedur activation pada kelas Radial_Basis_units.

```
void Radial_Basis_Topology::transfer_Gaussian_to_Output_layer(float percent, int test,
int cluster)
{
 int champ;
 float max = MINFLOAT;

 if(test == 1)
 {
 for(int i = 0; i < maximum_number_of_clusters; i++)
 {
 node_in_cluster_layer[i].activation() ;
 node_in_cluster_layer[i].Gaussian_transfer_output = exp( (-1.0) *
(node_in_cluster_layer[i].output_value));;
 }

 for(int i = 0; i < maximum_number_of_clusters; i++)
  if(node_in_cluster_layer[i].Gaussian_transfer_output > max)
  {
   max = node_in_cluster_layer[i].Gaussian_transfer_output;
   champ = i;
  }

 if(max < percent)
  for(int i = 0; i < maximum_number_of_clusters; i++)
  {
```

```
      node_in_cluster_layer[i].Gaussian_transfer_output = 0;
      node_in_output_layer.processing_unit_input[i] = 0;
    }
   else
    for(int i = 0; i < maximum_number_of_clusters; i++)
     if(i == champ)
     {
        node_in_cluster_layer[i].Gaussian_transfer_output = 1;
        node_in_output_layer.processing_unit_input[i] = 1;
     }
     else
     {
       node_in_cluster_layer[i].Gaussian_transfer_output = 0;
       node_in_output_layer.processing_unit_input[i] = 0;
     }
  }
  if(test == 0)
    for(int i = 0; i < maximum_number_of_clusters; i++)
    {
      node_in_cluster_layer[i].Gaussian_transfer_output = 0;
      node_in_output_layer.processing_unit_input[i] = 0;
      if(i == cluster)
      {
       node_in_cluster_layer[i].Gaussian_transfer_output = 1;
       node_in_output_layer.processing_unit_input[i] = 1;
      }
    }

  // transfer signal from cluster to output units and calculate output
  node_in_output_layer.calculate_output_signal(test);

}
```

d. Pada saat proses pelatihan output yang dikeluarkan oleh jaringan dibandingkan dengan target output yang diharapkan sesuai dengan data pelatihan. Dari pembandingan tersebut dapat dihasilkan nilai kesalahan yang terjadi yang nantinya akan digunakan untuk memperbaharui nila bobot dan bias dari jaringan.

```
void Radial_Basis_Topology::calculate_output_error_information_term(float target_value)
{
 float output_signal_derivative = calculate_output_signal_derivative();
 error_information_term = (target_value - output_signal) * output_signal_derivative;
 error_difference_squared = pow((target_value - output_signal), 2.0);}
```

e. Setelah proses pelatihan jaringan selesai parameter hasil pelatihan harus disimpan supaya saat proses pengenalan dilakukan tidak harus selalu dilakukan proses pelatihan terlebih dahulu.

```
void Radial_Basis_Topology::savenet(char *file)
{
 ofstream save_ptr;
 int node, dim;

 save_ptr.open(file, ios::out);

 save_ptr << 1 << "\n"; //network identifier number
 save_ptr << dimensions_of_signal << "\n";
 save_ptr << maximum_number_of_clusters << "\n";
 save_ptr << node_in_output_layer.bias << "\n";

 for(dim = 0; dim < maximum_number_of_clusters; dim++)
   save_ptr << node_in_output_layer.weight_of_inputs[dim] << " ";
 save_ptr << "\n";

 for(node = 0; node < maximum_number_of_clusters; node++)
  for(dim = 0; dim < dimensions_of_signal; dim++)
     save_ptr << node_in_cluster_layer[node].variance[dim] << " ";
  save_ptr << "\n";

 for(node = 0; node < maximum_number_of_clusters; node++)
 {
  for(dim = 0; dim < dimensions_of_signal; dim++)
  {save_ptr << node_in_cluster_layer[node].Mean[dim] << " ";}
  save_ptr << "\n";
 }
 save_ptr << MinOutput << "\n";
 save_ptr << MaxOutput;
 save_ptr.close();

}
```

f. Saat proses pengenalan pola citra wajah oleh jaringan FBR, parameter jaringan hasil pelatihan jaringan sebelumnya yang disimpan dalam bentuk file jaringan yang bertipe *.net dapat digunakan kembali dengan membuka file pelatihan tersebut.

```
void Radial_Basis_Topology::upload_network(char *getname)
{
 ifstream get_ptr;
 int netid, node, dim;
```

```
 int dolock = 0;
 TMsgDlgButtons Buttons;

 do
 {
  get_ptr.open(getname, ios::in);
  get_ptr >> netid;
  if(netid == 1) {dolock = 1;}
  else
  {
   Buttons << mbOK;
   int type = MessageDlg("Error** file contents do not match FBR
specifications\ntry again", mtWarning, Buttons, 0);
   get_ptr.close();
  }
 } while(dolock <= 0);
 get_ptr >> dimensions_of_signal;
 get_ptr >> maximum_number_of_clusters;

 node_in_output_layer.number_of_input_units = maximum_number_of_clusters;
 node_in_output_layer.establish_array_of_processing_unit_inputs();
 node_in_output_layer.establish_weight_vector_for_processing_units();
 get_ptr >> node_in_output_layer.bias;

 for(dim = 0; dim < maximum_number_of_clusters; dim++)
  get_ptr >> node_in_output_layer.weight_of_inputs[dim];

 node_in_cluster_layer = new Radial_Basis_units[maximum_number_of_clusters];
 for(node = 0; node < maximum_number_of_clusters; node++)
 {
  node_in_cluster_layer[node].number_of_inputs = dimensions_of_signal;
  node_in_cluster_layer[node].establish_input_output_arrays();
  node_in_cluster_layer[node].establish_Mean_array();
 }

 for(node = 0; node < maximum_number_of_clusters; node++)
 {
  for(dim = 0; dim < dimensions_of_signal; dim++)
  {get_ptr >> node_in_cluster_layer[node].variance[dim];}
 }

 for(node = 0; node < maximum_number_of_clusters; node++)
 {
  for(dim = 0; dim < dimensions_of_signal; dim++)
  {get_ptr >> node_in_cluster_layer[node].Mean[dim];}
 }
 get_ptr >> MinOutput;
 get_ptr >> MaxOutput;

 get_ptr.close();
}
```

4. Kelas Processing_units

a. Prosedur berikut melakukan perhitungan yang menghasilkan output yang dihasilkan oleh jaringan baik selama proses pelatihan maupun pada saat proses pengenalan pola.

```
Void Processing_units::calculate_output_signal(int test)
{
 sum_of_weighted_inputs = 0.0;
 for(int i = 0; i < number_of_input_units; i++)
 {
  if(i == number_of_input_units - 1)
  {sum_of_weighted_inputs += (processing_unit_input[i] * weight_of_inputs[i]) +
bias;}
  else
  {sum_of_weighted_inputs += processing_unit_input[i] * weight_of_inputs[i];}
 }
// binary sigmoid function
  output_signal = 1.0 / (1.0 + exp(-1.0 * sum_of_weighted_inputs));
  if(((sum_of_weighted_inputs - bias) == 0) && (test == 1) )
   output_signal = 0.0;
}
```

b. Proses perbaharuan nilai bobot dan bias dikerjakan oleh prosedur di bawah ini:

```
void Processing_units::calculate_weight_and_bias_correction_terms(float learning_rate)
{
 for(int i = 0; i < number_of_input_units; i++)
 {weight_correction_term[i] = learning_rate * error_information_term * processing_unit_input[i];}
 bias_correction_term = learning_rate * error_information_term;
 error_information_term = 0.0;
 update_weights_and_biases();
}

void Processing_units::update_weights_and_biases(void)
{
 for(int i = 0; i < number_of_input_units; i++)
 {weight_of_inputs[i] = weight_of_inputs[i] + weight_correction_term[i];}
 bias = bias + bias_correction_term;
}
```

5. Kelas Radial_Basis_units

a. Sebagai pembanding apakah suatu pola memiliki kemiripan antara yang satu dengan yang lain maka dibutuhkan nilai pembanding yaitu jarak euclidean yang dihitung oleh prosedur berikut ini.

```
Void Radial_Basis_units::calculate_sum_square_Euclidean_distance(void)
{
 double sumsquare;
 float ss1;
 int ci;
 output_value = 0.0;
 for(int k = 0; k < number_of_inputs; k++)
 {
  ci = k;

  if(input_value[ci] == 0.0)
  {
   sumsquare = pow(Mean[ci], 2.0);
  }
  else
  {
   sumsquare = pow(fabs(Mean[ci] - input_value[ci]), 2.0);
  }
  output_value += sumsquare;
 }
 ss1 = output_value;

 output_value = sqrt(fabs(ss1));

}
```

b. Prosedur activation bersama-sama dengan prosedur transfer_Gaussian_to_Output_layer pada kelas Radial_Basis_ topology melakukan perhitungan fungsi aktifasi pada tiap-tiap cluster yang terdapat di layer hidden.

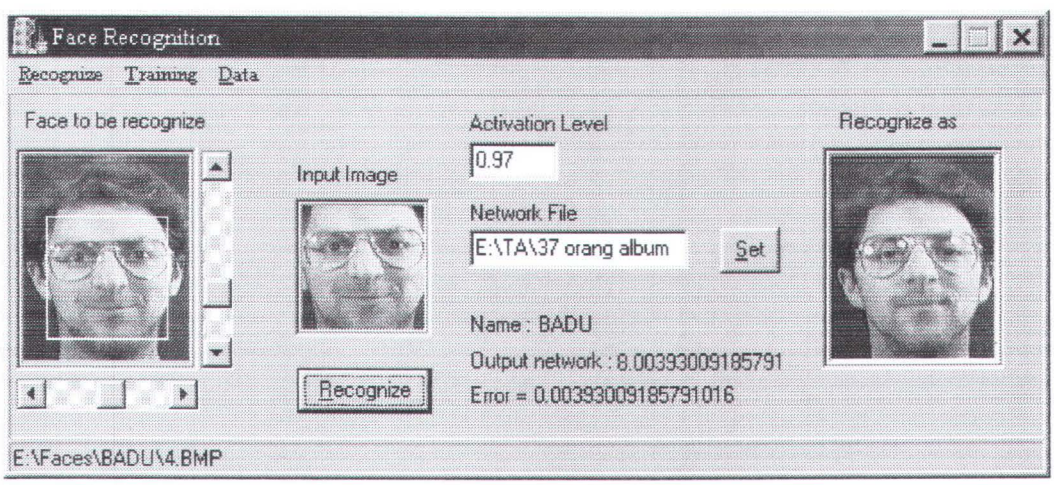
```
void Radial_Basis_units::activation(void)
{
 double sumsquare;
 output_value = 0.0;
 for(int k = 0; k < number_of_inputs; k++)
 {
  if(variance[k] == 0.0)
   sumsquare = pow(Mean[k], 2.0)/(2*0.02*15000);
  else
```

```
    sumsquare = pow(input_value[k]-Mean[k], 2 /  (2*15000*variance[k]);
   output_value += sumsquare;
  }
}
```

4.3.3 Implementasi Antar Muka

Seperti yang telah dijelaskan pada sub bab perancangan antar muka terdapat tiga menu utama yaitu: menu utama, menu pengumpulan data dan menu pengambilan data pelatihan.

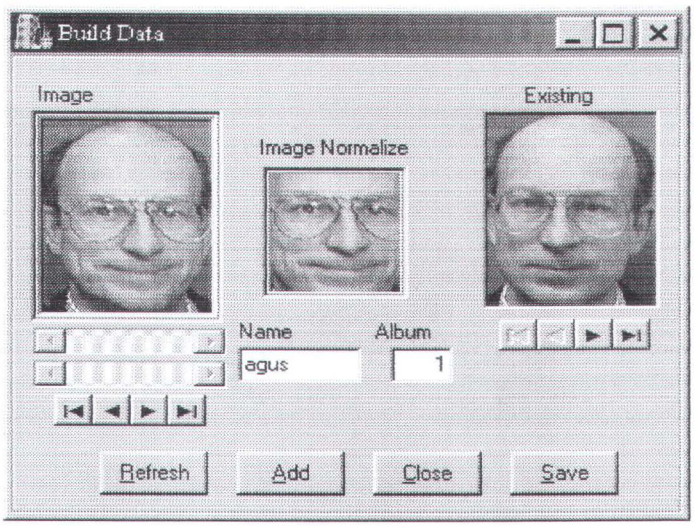
Tampilan antar muka untuk menu utama, dalam sistem ini, dapat dilihat pada Gambar 4.17 berikut ini :



Gambar 4.1 Form menu utama

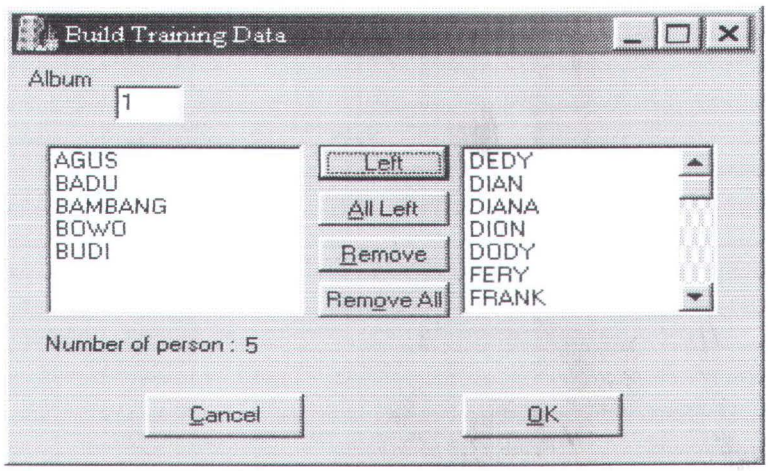
Pada form di atas, data – data yang perlu dimasukkan adalah citra wajah yang akan dikenali, file jaringan dan nilai aktifasi yang digunakan sebagai filter pengenalan system. Nilai yang terdapat pada variabel 'Output Network' merupakan nilai keluaran dari sistem pengenalan wajah dengan error pengenalan sebesar nilai yang terdapat pada variabel 'Error'.

Tampilan antar muka untuk menu pengumpulan data dapat dilihat pada Gambar 4.18. Di dalam menu ini proses pengumpulan data pelatihan dilakukan. Proses yang dilakukan adalah memilih dari sejumlah citra wajah yang ada kemudian data citra terpilih tersebut dimasukkan kedalam basis data wajah.



Gambar 4.2 Form pengumpulan data

Pada Gambar 4.19 merupakan implementasi dari menu pemilihan data pelatihan. Kolom di bagian kanan merupakan kumpulan citra wajah yang terdapat di dalam basis data sedangkan kolom di sebelah kiri menunjukkan kumpulan citra wajah terpilih dari basis data yang akan digunakan sebagai data pelatihan .



Gambar 4.3 Form menu pemilihan data pelatihan

BAB V

HASIL UJI COBA DAN EVALUASI

BAB V

HASIL UJI COBA DAN EVALUASI

Dalam bab ini dijelaskan tentang uji coba yang dilakukan terhadap perangkat lunak pengenalan wajah dengan menggunakan JST-FBR, serta pembahasan dan evaluasi yang diberikan berdasarkan hasil uji coba tersebut. Uji coba yang dilakukan terhadap perangkat lunak ini mempunyai tujuan untuk mengetahui kemampuan dan kelemahan sistem yang telah dibuat dan hasil dari pengujian tersebut akan digunakan sebagai dasar pertimbangan evaluasi terhadap perangkat lunak pengenalan wajah ini. Hasil evaluasi yang diperoleh dapat digunakan untuk pengembangan sistem pengenalan wajah lebih lanjut.

5.1 LINGKUNGAN UJI COBA

Spesifikasi perangkat lunak yang digunakan dalam uji coba perangkat lunak ini adalah sebagai berikut :

- Processor Intel Celeron 400 MMX
- RAM 128 MB
- Hardisk 6.4 GB

Spesifikasi lain yang juga digunakan dan turut menunjang uji coba ini dan pembuatan perangkat lunak ini adalah :

- Sistem operasi Windows 98
- Borland C++ Builder 4.0

5.2 DATA UJI COBA

Citra wajah yang digunakan pada pelaksanaan uji coba perangkat lunak pengenalan wajah ini mempunyai ukuran 92x112 piksel. Citra wajah yang digunakan adalah citra *gray-scale* bertipe bitmap. Data citra wajah tersebut didapat dari basis data wajah laboratorium riset Olivetti. Basis data ini terdiri dari 370 model citra wajah dari 37 orang dengan asumsi 10 variasi untuk tiap orang. Separuh dari data ini digunakan untuk pelatihan jaringan dan separuh sisanya digunakan sebagai data uji coba. Tiap-tiap data dari tiap individu mempunyai nomor index 1 sampai 10.

5.3 PELAKSANAAN UJI COBA

Pengujian yang dilakukan terhadap sistem pengenalan wajah ini menggunakan dua tipe data dengan 5 data pelatihan untuk tiap individu. Sebelum data citra tersebut diproses lebih lanjut, citra yang hendak diproses atau dikenali oleh sistem direduksi dimensinya terlebih dahulu. Dengan dilakukan pre-proses, citra masukkan akan berukuran 20x20 piksel. Diharapkan dari pre-proses ini akan mempercepat waktu pelatihan jaringan.

Kedua tipe data pelatihan tersebut sebagai berikut :

a) Data pelatihan acak:

Lima data fitur wajah ternormalisasi diambil secara urut dari indeks nomor 1 sampai dengan nomor 5 dari variasi yang ada. Sedangkan sisanya digunakan sebagai uji coba sistem.

b) Data pelatihan terpilih

Lima data fitur wajah ternormalisasi dipilih dari 10 variasi berdasarkan pertimbangan variasi data wajah yang ada untuk tiap individu.

5.3.1 Uji Coba Dengan Data Pelatihan Acak

Pada uji coba ini jaringan syaraf tiruan pengenal wajah dilatih dengan menggunakan lima data pelatihan untuk tiap individu yang didapat secara acak berdasarkan urutan index. Sehingga terdapat 185 data untuk pelatihan jaringan. Jumlah cluster yang terdapat pada layer hidden berjumlah sesuai dengan data pelatihan. Pada pengujian ini digunakan 4 level aktivasi yang berbeda sebagai parameter pengenalan untuk cluster pada layer hidden dengan nilai 0.95, 0.96, 0.97 dan 0.98. Parameter ini diambil berdasarkan kenyataan bahwa tiap-tiap unit hidden yang terasosiasi dengan sebuah citra, yang dilakukan proses pengenalan, idealnya memberikan nilai 1, dan unit hidden yang tidak berasosiasi bernilai 0. Sehingga level aktivasi diberikan nilai yang mendekati nilai optimal. Waktu yang digunakan untuk proses pelatihan jaringan pengenal wajah ini memakan waktu kurang dari 5 menit dengan data dari 37 individu. Untuk pengenalan satu citra wajah membutuhkan waktu kurang dari 1 detik.

Hasil yang dicapai dari uji coba sebagai berikut :

- Untuk level aktivasi 0.95, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 159 dari 185 data atau sekitar 85,95% seperti yang terlihat pada Tabel 5.1.

- Untuk level aktivasi 0.96, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 151 dari 185 data atau sekitar 81,2% seperti yang terlihat pada Tabel 5.2.
- untuk level aktivasi 0.97, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 142 citra wajah uji coba atau sekitar 76,76% seperti yang terlihat pada Tabel 5.3.
- untuk level aktivasi 0.98, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 106 citra wajah dari 185 citra wajah yang digunakan sebagai data pelatihan. Persentase dari keberhasilan pengenalan ini adalah 57,3%. Lihat Tabel 5.4.

5.3.2 Uji Coba Dengan Data Pelatihan Terpilih

Dilihat dari hasil uji coba sebelumnya, untuk uji coba dengan data pelatihan tanpa pre-proses tidak dilaksanakan disini. Hal ini didasarkan atas pertimbangan kecepatan pelatihan jaringan syaraf tiruan pengenal wajah. Untuk data pelatihan tanpa pre-proses, proses pelatihan memakan waktu sampai 3 jam sedangkan dengan data pelatihan yang telah mengalami pre-proses, waktu pelatihan dapat diselesaikan dalam hitungan kurang dari 5 menit.

Pada uji coba ini jaringan syaraf tiruan pengenal wajah dilatih dengan menggunakan lima data pelatihan untuk tiap individu yang didapat berdasarkan pertimbangan variasi data yang ada tanpa memperhatikan urutan indeksnya. Sehingga terdapat 185 data untuk pelatihan jaringan. Variabel yang terdapat pada pelatihan jaringan syaraf tiruan pengenal wajah untuk uji coba ini bernilai sama dengan uji coba sebelumnya yaitu dengan menggunakan 4 level aktivasi yang

berbeda sebagai parameter pengenalan untuk cluster pada layer hidden dengan nilai 0.95, 0.96, 0.97 dan 0.98. Proses pengenalan yang dijalankan pada uji coba ini memakan waktu kurang dari 1 detik.

Hasil yang dicapai dari uji coba sebagai berikut :

- Untuk level aktivasi 0.95, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 176 atau sekitar 95,13% seperti yang terlihat pada Tabel 5.5.
- Untuk level aktivasi 0.96, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 181 atau sekitar 97,84% seperti yang terlihat pada Tabel 5.6.
- untuk level aktivasi 0.97, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 181 citra wajah uji coba atau sekitar 97,84% seperti yang terlihat pada Tabel 5.7.
- untuk level aktivasi 0.98, jaringan pengenal wajah berhasil mendeteksi citra wajah dengan tepat sebanyak 159 citra wajah dari 185 citra wajah yang digunakan sebagai data pelatihan. Persentase dari keberhasilan pengenalan ini adalah 85,95%. Lihat Tabel 5.8.

Tabel 5.1 Hasil uji coba dengan data pelatihan acak, nilai aktivasi 0.95

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	**4**	**0**	**1**	5
6	5	0	0	5
7	5	0	0	5
8	**3**	**0**	**2**	5
9	**4**	0	**1**	5
10	**4**	**0**	**1**	5
11	**4**	0	**1**	5
12	**4**	**0**	**1**	5
13	**2**	**0**	**3**	5
14	5	0	0	5
15	5	0	0	5
16	**4**	**0**	**1**	5
17	5	0	0	5
18	5	0	0	5
19	**2**	**0**	**3**	5
20	5	0	0	5
21	5	0	0	5
22	5	0	0	5
23	**4**	**0**	**1**	5
24	5	0	0	5
25	5	0	0	5
26	**4**	**0**	**1**	5
27	5	0	0	5
28	**1**	**0**	**4**	5
29	**4**	**0**	**1**	5
30	5	0	0	5
31	5	0	0	5
32	5	0	0	5
33	5	0	0	5
34	5	0	0	5
35	5	0	0	5
36	5	0	0	5
37	**2**	**0**	**3**	5
Total	**159**	**0**	**26**	**185**
Prosentase Keberhasilan			**85.95%**	

Tabel 5.2 Hasil uji coba dengan data pelatihan acak, nilai aktivasi 0.96

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	**4**	**0**	**1**	5
6	5	0	0	5
7	5	0	0	5
8	**2**	**0**	**3**	5
9	5	0	0	5
10	**4**	**0**	**1**	5
11	5	0	0	5
12	**4**	**0**	**1**	5
13	**3**	**0**	**2**	5
14	**1**	**3**	**1**	5
15	5	0	0	5
16	**3**	**0**	**2**	5
17	5	0	0	5
18	5	0	0	5
19	**2**	**2**	**1**	5
20	5	0	0	5
21	5	0	0	5
22	5	0	0	5
23	**2**	**2**	**1**	5
24	5	0	0	5
25	5	0	0	5
26	**3**	**0**	**2**	5
27	5	0	0	5
28	**0**	**0**	**5**	5
29	**3**	**0**	**2**	5
30	5	0	0	5
31	**4**	**0**	**1**	5
32	5	0	0	5
33	5	0	0	5
34	**4**	**1**	**0**	5
35	5	0	0	5
36	5	0	0	5
37	**2**	**0**	**3**	5
Total	**151**	**8**	**26**	**185**
Prosentase Keberhasilan			**81.62%**	

Tabel 5.3 Hasil uji coba dengan data pelatihan acak, nilai aktivasi 0.97

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi Salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	5	0	0	5
6	**4**	**1**	**0**	5
7	**4**	**0**	**0**	5
8	**2**	**2**	**1**	5
9	**4**	**0**	**1**	5
10	**4**	**1**	**0**	5
11	**4**	**1**	**0**	5
12	**4**	**0**	**1**	5
13	**3**	**1**	**1**	5
14	**0**	**5**	**0**	5
15	5	0	0	5
16	**3**	**0**	**2**	5
17	5	0	0	5
18	**4**	**1**	**0**	5
19	**1**	**4**	**0**	5
20	5	0	0	5
21	5	0	0	5
22	5	0	0	5
23	**2**	**2**	**1**	5
24	5	0	0	5
25	5	0	0	5
26	**3**	**2**	**0**	5
27	**4**	**1**	**0**	5
28	**0**	**2**	**3**	5
29	**3**	**1**	**1**	5
30	5	0	0	5
31	**4**	**1**	**0**	5
32	5	0	0	5
33	5	0	0	5
34	**4**	**1**	**0**	5
35	5	0	0	5
36	**4**	**1**	**0**	5
37	**1**	**3**	**1**	5
Total	**142**	**30**	**12**	**185**
Prosentase keberhasilan			**76,76%**	

Tabel 5.4 Hasil uji coba dengan data pelatihan acak, nilai aktivasi 0.98

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	3	2	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	2	3	0	5
6	4	1	0	5
7	3	2	0	5
8	2	3	0	5
9	2	3	0	5
10	2	3	0	5
11	4	1	0	5
12	3	2	0	5
13	3	1	1	5
14	0	5	0	5
15	4	1	0	5
16	3	2	0	5
17	3	2	0	5
18	3	2	0	5
19	0	5	0	5
20	4	1	0	5
21	4	1	0	5
22	5	0	0	5
23	1	4	0	5
24	4	1	0	5
25	0	5	0	5
26	3	2	0	5
27	2	3	0	5
28	0	5	0	5
29	3	2	0	5
30	5	0	0	5
31	3	2	0	5
32	3	2	0	5
33	2	3	0	5
34	2	3	0	5
35	5	0	0	5
36	3	2	0	5
37	1	478	0	5
Total	**106**	**78**	**1**	**185**
Prosentase keberhasilan			**57.3%**	

Tabel 5.5 Hasil uji coba dengan data pelatihan terpilih, nilai aktivasi 0.95

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	5	0	0	5
6	5	0	0	5
7	5	0	0	5
8	5	0	0	5
9	5	0	0	5
10	5	0	0	5
11	**4**	0	**1**	5
12	5	0	0	5
13	5	0	0	5
14	5	0	0	5
15	5	0	0	5
16	5	0	0	5
17	5	0	0	5
18	5	0	0	5
19	**2**	**0**	**3**	5
20	5	0	0	5
21	5	0	0	5
22	5	0	0	5
23	5	0	0	5
24	5	0	0	5
25	5	0	0	5
26	**3**	**0**	**2**	5
27	5	0	0	5
28	**4**	**0**	**1**	5
29	5	0	5	5
30	5	0	0	5
31	5	0	0	5
32	5	0	0	5
33	5	0	0	5
34	5	0	0	5
35	5	0	0	5
36	5	0	0	5
37	**4**	**0**	**1**	5
Total	**176**	**0**	**9**	**185**
Prosentase Keberhasilan			**95.13%**	

Tabel 5.6 Hasil uji coba dengan data pelatihan terpilih, nilai aktivasi 0.96

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	5	0	0	5
6	5	0	0	5
7	5	0	0	5
8	5	0	0	5
9	5	0	0	5
10	5	0	0	5
11	**4**	**0**	**1**	5
12	5	0	0	5
13	5	0	0	5
14	5	0	0	5
15	5	0	0	5
16	5	0	0	5
17	5	0	0	5
18	5	0	0	5
19	**2**	**0**	**3**	5
20	5	0	0	5
21	5	0	0	5
22	5	0	0	5
23	5	0	0	5
24	5	0	0	5
25	5	0	0	5
26	5	0	0	5
27	5	0	0	5
28	5	0	0	5
29	5	0	0	5
30	5	0	0	5
31	5	0	0	5
32	5	0	0	5
33	5	0	0	5
34	5	0	0	5
35	5	0	0	5
36	5	0	0	5
37	5	0	0	5
Total	**181**	**0**	**4**	**185**
Prosentase keberhasilan			**97,84%**	

Tabel 5.7 Hasil uji coba dengan data pelatihan terpilih, nilai aktivasi 0.97

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	5	0	0	5
4	5	0	0	5
5	5	0	0	5
6	5	0	0	5
7	5	0	0	5
8	5	0	0	5
9	5	0	0	5
10	5	0	0	5
11	**4**	**0**	**1**	5
12	5	0	0	5
13	5	0	0	5
14	5	0	0	5
15	5	0	0	5
16	5	0	0	5
17	5	0	0	5
18	5	0	0	5
19	**2**	**2**	**1**	5
20	5	0	0	5
21	5	0	0	5
22	5	0	0	5
23	5	0	0	5
24	5	0	0	5
25	5	0	0	5
26	5	0	0	5
27	5	0	0	5
28	5	0	0	5
29	5	0	0	5
30	5	0	0	5
31	5	0	0	5
32	5	0	0	5
33	5	0	0	5
34	5	0	0	5
35	5	0	0	5
36	5	0	0	5
37	5	0	0	5
Total	**181**	**2**	**2**	**185**
Prosentase keberhasilan			**97,84%**	

Tabel 5.8 Hasil uji coba dengan data pelatihan terpilih, nilai aktivasi 0.98

Pola	Jumlah keberhasilan	Tak terdeteksi	Terdeteksi salah	Jumlah Data
1	5	0	0	5
2	5	0	0	5
3	4	1	0	5
4	5	0	0	5
5	3	2	0	5
6	5	0	0	5
7	3	0	0	5
8	5	0	0	5
9	5	0	0	5
10	5	0	0	5
11	3	2	0	5
12	4	1	0	5
13	5	0	0	5
14	5	0	0	5
15	5	0	0	5
16	4	1	0	5
17	5	0	0	5
18	4	1	0	5
19	3	3	0	5
20	5	0	0	5
21	4	1	0	5
22	5	0	0	5
23	5	0	0	5
24	5	0	0	5
25	4	1	0	5
26	2	3	0	5
27	1	4	0	5
28	4	1	0	5
29	4	1	0	5
30	5	0	0	5
31	5	0	0	5
32	5	0	0	5
33	3	2	0	5
34	4	1	0	5
35	5	0	0	5
36	5	0	0	5
37	5	0	0	5
Total	159	26	0	185
Prosentase keberhasilan			85,95%	

5.4 EVALUASI HASIL UJI COBA

Uji coba yang telah dilakukan terhadap jaringan syaraf tiruan pengenalan wajah ini menunjukkan hasil yang bervariasi tergantung dari pemilihan data dan level aktivasi yang digunakan. Prosentase keberhasilan pengenalan terbesar yang dicapai adalah 97,84% dengan menggunakan level aktivasi 0.96 dan 0.97 dari data pelatihan yang telah dipilih terlebih dahulu berdasarkan variasi data yang ada.

Pemilihan data pelatihan yang mewakili semua variasi atau mendekati semua variasi dalam data uji coba memberikan hasil pengenalan yang meningkat jika dibandingkan dengan mengambil data pelatihan secara acak tanpa memperhatikan variasi data yang ada. Tabel berikut memberikan perbandingan tingkat keberhasilan dari pemilihan data dari uji coba yang telah dilakukan.

Tabel 5.1 Perbandingan keberhasilan pengenalan

Pemilihan data pelatihan	*Level Aktivasi*			
	0.95	0.96	0.97	0.98
Pemilihan acak	85,95%	81,62%	76,76%	57,3%
Pemilihan berdasarkan variasi	95,13%	97,84%	97,84%	85,95%

Dari Tabel 5-1 sampai 5-6 dapat diamati frekwensi kesalahan yang terjadi dengan penggunaan level aktivasi yang berbeda, berikut ini merupakan tabel perbandingan berdasarkan jumlah kesalahan pengenalan yang terjadi terhadap level aktivasi.

Tabel 5.2 Frekwensi berdasarkan level aktivasi untuk data pelatihan acak

Kategori kesalahan	*Level Aktivasi*			
	0.95	0.96	0.97	0.98
Tidak terdeteksi	0	8	30	78
Salah mendeteksi	26	26	12	1
Total	26	34	42	79

Tabel 5.3 Frekwensi berdasarkan level aktivasi untuk data pelatihan terpilih

Kategori kesalahan	*Level Aktivasi*			
	0.95	0.96	0.97	0.98
Tidak terdeteksi	0	0	2	26
Salah mendeteksi	9	4	2	0
Total	9	4	4	26

Dengan memperhatikan tabel-tabel sebelumnya, dapat diketahui bahwa jika nilai level aktivasi dipertinggi maka frekwensi kesalahan pengenalan atau pengenalan negatif yang dilakukan oleh jaringan syaraf tiruan pengenalan wajah semakin mengecil dan prosentase citra tidak dapat dikenali semakin besar. Bentuk dari kesalahan pengenalan ini adalah sistem mendeteksi orang yang salah dari yang seharusnya. Namun terdapat batas dimana efek dari mempertinggi nilai level aktifasi akan menurunkan prosentase keberhasilan pengenalan oleh jaringan syaraf pengenalan wajah secara keseluruhan.

BAB VI

PENUTUP

BAB VI

PENUTUP

6.1 KESIMPULAN

Proses pengenalan wajah dengan menggunakan Jaringan Syaraf Tiruan Fungsi Basis Radial pada perangkat lunak ini dapat diambil beberapa kesimpulan, yaitu :

1. Perangkat lunak pengenalan wajah berbasis Jaringan Syaraf Tiruan Fungsi Basis Radial dapat digunakan untuk mengenali citra wajah manusia yang berupa citra dua dimensi. Perangkat lunak ini dapat mengenali citra wajah seseorang tanpa memperdulikan perubahan struktur muka yang diakibatkan oleh aksesoris dan emosi wajah.
2. Besarnya prosentase pengenalan perangkat lunak yang dibuat pada Tugas Akhir ini tergantung dari pemilihan data pelatihan. Semakin data pelatihan mewakili citra yang hendak dikenali maka semakin tinggi prosentase pengenalannya. Prosentase pengenalan terbesar dicapai oleh sistem dengan menggunakan data pelatihan terpilih sebesar 97,84%. Prosentase kesalahan pengenalan yang terjadi, untuk data pelatihan yang acak lebih besar jika dibandingkan dengan prosentase hasil yang diperoleh dengan menggunakan data pelatihan yang dipilih menurut variasi data yang diwakilinya.
3. Nilai aktivasi yang digunakan saat proses pengenalan sangat berpengaruh terhadap hasil pengenalan yang didapat. Dengan menggunakan nilai aktifasi

0.97, untuk data pelatihan yang dipilih tanpa memperdulikan variasi data yang diwakili, didapat nilai prosentase pengenalan terbesar sebanyak 85,95% keberhasilan. Sedangkan untuk data pelatihan yang dipilih berdasarkan variasi data yang diwakilinya dengan menggunakan nilai aktivasi 0.96 dan 0.97, sistem pengenalan wajah ini menghasilkan nilai prosentase pengenalan terbesar sebanyak 97,84%. Untuk proses pengenalan dengan menggunakan data pelatihan yang telah dipilih berdasarkan variasi data yang diwakilinya, semakin besar nilai aktivasi maka prosentase pengenalan negatif (yaitu terdeteksi salah) semakin kecil. Sebaliknya apabila nilai aktivasi semakin kecil maka prosentase pengenalan negatif akan semakin besar.

4. Waktu proses saat pelatihan dan pengenalan yang dilakukan pada perangkat lunak pengenalan wajah ini berlangsung relatif singkat. Besarnya waktu yang dibutuhkan saat proses pelatihan, sebanding dengan jumlah data pelatihan yang digunakan. Sedangkan besarnya waktu yang dibutuhkan untuk proses pengenalan dipengaruhi oleh jumlah node pada hidden layer yang menyusun jaringan.

6.2 KEMUNGKINAN PENGEMBANGAN LEBIH LANJUT

Pengembangan yang mungkin dapat dilakukan untuk perangkat lunak pengenalan wajah yang dibuat pada tugas akhir ini adalah penggunaan otomatisasi pada pengambilan citra normalisasi saat proses ekstraksi fitur yang pada Tugas Akhir ini masih dilakukan secara manual. Dengan menambahkan otomatisasi tersebut diharapkan kesalahan penggunaan perangkat lunak saat proses

pengambilan citra normalisasi dapat dikurangi atau bahkan dapat dihilangkan sehingga prosentase keakuratan pengenalan dapat ditingkatkan.

DAFTAR PUSTAKA

DAFTAR PUSTAKA

[BAR-92] Bart Kosco, *"Neural Networks and Fuzzy Systems"*, Prentice Hall, 1992

[GUT-95] Gutta, S., J. Huang, D. Singh, I. Shah, B. Takacs, and H. Wechsler, "Benchmark Studies on Face Recognition", *Proceedings of International Workshop on Automatic Face – and Gesture Recognition (IWAFGR)*, Zurich, Switzerland, June 1995.

[JON-96] A. Jonathan Howell and Hilary Buxton, *"Face Recognition using Radial Basis Function Neural Networks"*, School of Cognitive and Computing Sciences, University of Sussex, Falmer, Brighton BN1 9QH, UK, 1996.

[JWN-96] Jose C. Principe, W. Curt Lefebvre and Neil R. Euliano, *"Neural Systems: Fundamentals Through Simulation"*, Principe 1996, http://www.cnel.ufl.edu.

[KRO-95] Krogh, A., and Vedelsby, J., *"Neural Network Ensembles"*, Cross Validation and Active Learning, *NIPS*, 7, Morgan Kaufmann, 1995

[LES-98] Dr. Leslie Smith, *"An Introduction to Neural Networks"*, Centre for Cognitive and Computational Neuroscience, Department of Computing and Mathematics, http://www.cs.stir.ac.uk/~lss/, 1998.

[LMS-97] The Least Mean Square (LMS) Algorithm, http://dspserv.sdsu.edu/, 1997

[NFR] Neural Networks for Face Recognition, http://cs.cmu.edu/.

[SEU-92] Seung, H. S. *et. al.* "Query by Committee", in *Proceedings of the fifth Workshop on Computational Learning Theory,* 5, 287-294, San Mateo, Ca, Morgan Kaufmann, 1992.

[SRI-97] Srinivas G., and Harry W., *"Face Recognicion Using Hybrid Classifier Systems"*, Department of Computer Science, George Manson University, http://chagall.gmu.edu/FORENSIC, 1997.

[SUB-99] Subhan Agus Akhbar, "Perancangan dan Pembuatan Perangkat Lunak Pengenalan Wajah dengan Menggunakan Jaringan Syaraf yang Didasarkan pada Keputusan Probabilistik", *Tugas Akhir*, Jurusan Teknik Informatika, Fakultas Teknologi Industri Institut, Teknologi Sepuluh Nopember Surabaya 1999.

[TAK-95] Takaes B., and H. Wechsler, "Face Location Using a Dynamic Model of Retinal Feature Extraction", *Proceedings of International Workshop on Automatic Face – and Gesture Recognition (IWAFGR)*, Zurich, Switzerland, June 1995.