Face Detection Using Backpropagation Neural Networks

Anifuddin Azis[1], Muhamad Haikal[2]

[1,2] Computer Science Study Program, Mathematics and Natural Science Faculty, Gadjah Mada University, Yogyakarta

E-mail: [1]anifudin@ugm.ac.id

Abstract

This paper summarizes a research effort in human face detection. A system to locate human faces in images, especially when used as a front-end for a human face identification system, could have many applications in the law enforcement and security professions. The approach presented here is a hybrid system using an edge detetion preprocessor and back-propagation neural networks. The method proposed successfully detected multiple faces. The results obtained are reported along with a discussion for improving the system.

Keywords: *Backpropagation neural networks, Edge Detection*

1. Pendahuluan

Teknik-teknik identifikasi wajah sudah banyak ditawarkan untuk menghasilkan sistem komputasi identifikasi wajah yang ideal. Teknik yang sering digunakan diantaranya adalah *Principal Component Analysis* (PCA), *Gabor Wavelet*, *Embedde Hidden Markov*, dan lain-lain. Teknik lain yang dapat digunakan adalah dengan deteksi tepi dengan konsep jaringan syaraf tiruan.

Deteksi tepi (*edge detection*) merupakan salah satu wilayah pengolahan citra digital yang paling awal dan paling banyak diteliti. Proses ini seringkali ditempatkan sebagai langkah pertama dalam aplikasi segmentasi citra, yang bertujuan untuk mengenali objek-objek yang terdapat dalam citra ataupun konteks citra secara keseluruhan. Deteksi tepi berfungsi untuk mengidentifikasi garis batas (*boundary*) dari suatu objek yang terdapat pada citra. Tepian dapat dipandang sebagai lokasi piksel dimana terdapat nilai perbedaan intensitas citra secara ekstrem. Idealnya, proses deteksi tepi akan menggambarkan bentuk geometris dari suatu obyek dan mengidentifikasi garis-garis yang mendasari obyek tersebut. *Output* dari operasi ini dimanfaatkan untuk pemerosesan visual yang lebih tinggi seperti rekonstruksi 3 dimensi, kompresi citra serta identifikasi obyek.

Dalam proses identifikasi dibutuhkan suatu metode untuk mempelajari *output* dari hasil deteksi tepi, yaitu dengan jaringan syaraf tiruan. Jaringan Syaraf Tiruan (JST) merupakan salah satu sistem pemrosesan informasi yang didesain dengan menirukan cara kerja otak manusia dalam menyelesaikan suatu masalah dengan melakukan proses belajar melalui perubahan bobot sinapsisnya. Jaringan syaraf tiruan memiliki kemampuan untuk mempelajari dependensi secara langsung dari data historis tanpa perlu memilih model yang cocok sehingga dapat memberikan keputusan terhadap data yang belum pernah di pelajari.

Konsep jaringan syaraf tiruan yang berkembang pesat, menimbulkan beberapa metode untuk menciptakan permodelan dari jaringan syaraf tiruan dengan kekurangan dan kelebihan masing-masing. Salah satu metode

dari jaringan syaraf tiruan adalah *Backpropagation* (Propagasi Balik).

1.1 Batasan Masalah

Beberapa batasan yang digunakan dalam penelitian adalah sebagai berikut :

1. Citra wajah yang digunakan adalah citra dengan ukuran 90x112 pixel dengan tipe bmp (Bitmap).
2. Deteksi tepi yang digunakan adalah deteksi tepi berbasis gradien dengan operator Roberts.
3. Dalam proses pelatihan dan pengujian digunakan 4 jenis data identitas dengan citra wajah yang dimiliki oleh masing-masing identitas.

1.2 Tujuan Penelitian

Tujuan dari penelitian ini adalah dapat merancang serta mengimplementasikan sistem dengan menggunakan deteksi tepi dan jaringan syaraf tiruan *backpropagation* untuk mengidentifikasi citra wajah.

2. Landasan Teori

2.1 Deteksi Tepi

Tepi adalah batas antara 2 daerah dengan sifat tingkat keabuan yang relatif berbeda [3]. Batas suatu obyek semakin nampak bersamaan dengan ketidak kontinuitasan intensitas dalam suatu citra. Eksperimen pada sistem pengeliahatan manusia menunjukkan bahwa suatu citra sangat penting, bahkan suatu obyek dapat dikenali hanya dari garis bentuk kasarnya saja (*croude ourline*). Kenyataan ini merupakan konsep prinsipil untuk merepresentasikan suatu obyek melalui batasnya. Lagipula, representasi batas mudah untuk diintegrasikan dengan berbagai macam-macam algoritma.

Tepi lokal (*local edge*) adalah daerah kecil dalam suatu citra yang tingkat keabuan lokalnya berubah dengan sangat cepat dengan cara yang sederhana (misalnya monotonic). Operator tepi (*edge operator*) adalah operator matematis (atau yang hasil komputasinya ekuivalen), yang dengan perluasan kecil, didisain untuk mendeteksi keberadaan tepi lokal dalam suatu fungsi citra [6]. Pada intinya, ide yang mendasari sebagian algoritma pendeteksian tepi adalah perhitungan suatu operator derivatif lokal.

2.2 Deteksi Tepi Berbasis Gradien

Dalam teknik deteksi tepi ini, diasumsikan bahwa tepi adalah pixel yang memiliki nilai gradien tinggi. Gradien itu sendiri adalah ukuran besarnya perubahan intensitas yang terjadi.

Pada citra digital $f(x,y)$, turunan berarah sepanjang tepian objek akan bernilai maksimum pada arah normal dari kontur tepian yang bersesuaian. Sifat ini dipergunakan sebagai dasar pemanfaatan operator gradien sebagai deteksi tepi (*edge detector*). Gradien suatu citra $f(x,y)$ pada lokasi (x, y) dapat didefinisikan sebagai vektor [3]:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (1)$$

Sudah dikenal dari analisis vektor, bahwa vektor G menunjukkan arah rerata maksimum dari perubahan f di lokasi (x, y). Untuk pendeteksi tepi, yang diperhatikan adalah besarnya vektor tersebut, yang umumnya dikenal sebagai gradien yang dinyatakan oleh ∇f, dimana:

$$\nabla f = mag(\nabla f) = \left[G_x^2 + G_y^2\right]^{1/2} \quad (2)$$

Kuantitas ini sama dengan rerata maksimum kenaikan $f(x, y)$ per-satuan jarak dengan arah ∇f. Persamaan diatas dapat dibuat lebih praktis lagi dengan mengaproksimasikan nilai gradien dengan nilai absolutnya.

$$\nabla f \approx |G_x| + |G_y| \quad (3)$$

Aproksimasi ini jauh lebih mudah untuk diimplementasikan, khususnya jika perangkat keras terdedikasi akan digunakan.

Arah vektor gradien juga adalah kuantitas yang penting. Jika $\alpha(x, y)$ merepresentasikan sudut arah ∇f pada lokasi (x, y), dari analisis vektor:

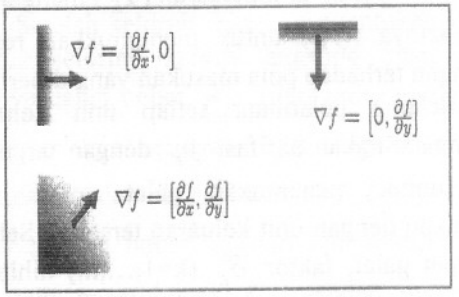
$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \quad (4)$$

Di mana sudut diukur dengan memperhatikan sumbu x. Persamaan 4 adalah alat yang berguna untuk menghubungkan titik tepi yang telah terdeteksi dengan menggunakan gradien.

Untuk konvolusi *mask* yang bersesuaian dinyatakan dengan

$$G_x \approx [-1 \;\; 1] \quad G_y \approx \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad (5)$$

Pada dasarnya *mask* merupakan larik 2 dimensi kecil yang koefisiennya dipilih untuk mendeteksi sifat tertentu dari suatu citra. Salah satu deteksi tepi berbasis garadien yang akan digunakan nantinya adalah operator Roberts.



Gambar 1 Arah gradien pada deteksi tepi

2.3 Operator Roberts

Roberts membuat *mask* untuk menerapkan prinsip di atas yang dinyatakan sebagai berikut

$$G_x \approx \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \; G_y \approx \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad (6)$$

Diferensi diambil secara diagonal untuk mandapatkan *mask* yang berbentuk bujursangkar, dan diperoleh 2 *mask* mengakomodasi kedua arah diagonal. Hasil konvolusi citra asal dengan kedua *mask* tersebut kemudian digabungkan dengan cara tertentu sehingga menghasilkan nilai yang dapat digunakan untuk menentukan apakah sebuah titik merupakan tepi obyek dalam citra. Biasanya tepi naik (hasil positif) dan tepi turun (hasil negatif) tidak dibedakan, sehingga diambil nilai absolutnya [1].

Karena ukuran *mask* Operator Roberts adalah 2x2, maka posisi titik yang ditinjau tidak berada di pusat *mask*, sehingga tepi yang dihasilkan berada pada salah satu sisi tepi tersebut (tepi atas atau tepi bawah).

Hasil konvolusi biasanya ada disekitar nilai 0 (nilai kecil), sehingga jika ditampilkan, citra tepi kebanyakan berwarna gelap (hitam), oleh karena itu diperlukan operasi negasi.

(a) citra asli

(b) hasil deteksi tepi

(c) hasil deteksi tepi yang dinegasikan

Gambar 2. Deteksi tepi dengan Operator Roberts

2.4 Jaringan Syaraf Tiruan

Pada tahun 1943, McCulloh dan Pitts merancang model formal sebagai perhitungan dasar *neuron* yang merupakan jaringan syaraf tiruan yang pertama kali. Jaringan syaraf tiruan sendiri adalah sistem pengolah informasi yang mempunyai karakteristik kerja yang identik dengan sistem kerja jaringan syaraf biologis manusia [2].

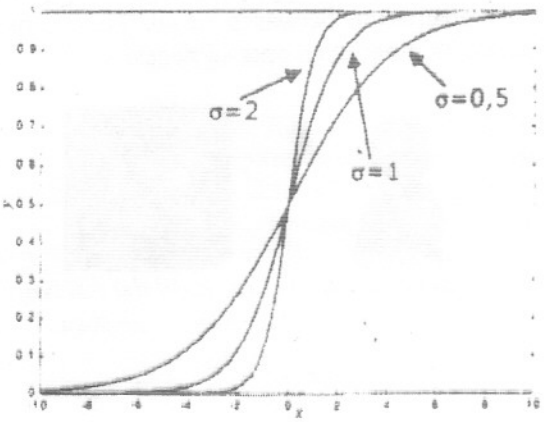
2.4.1 Fungsi Aktivasi Sigmoid Biner

Fungsi ini digunakan untuk jaringan syaraf yang dilatih dengan menggunakan metode *backpropagation*. Fungsi sigmoid biner memiliki nilai pada range 0 sampai 1. Oleh karena itu, fungsi ini sering digunakan untuk jaringan syaraf yang membutuhkan nilai yang terletak pada interval 0 sampai 1. Namun, fungsi ini bisa juga digunakan oleh jaringan syaraf yang nilai outputnya 0 atau 1.

Fungsi sigmoid biner dirumuskan sebagai :

$$y = f(x) = \frac{1}{1+e^{-\sigma x}} \qquad (7)$$

dengan : $f'(x) = \sigma f(x)[1 - f(x)]$



Gambar3 .Fungsi aktivasi : Sigmoid Biner

2.4.2 *Backpropagation*

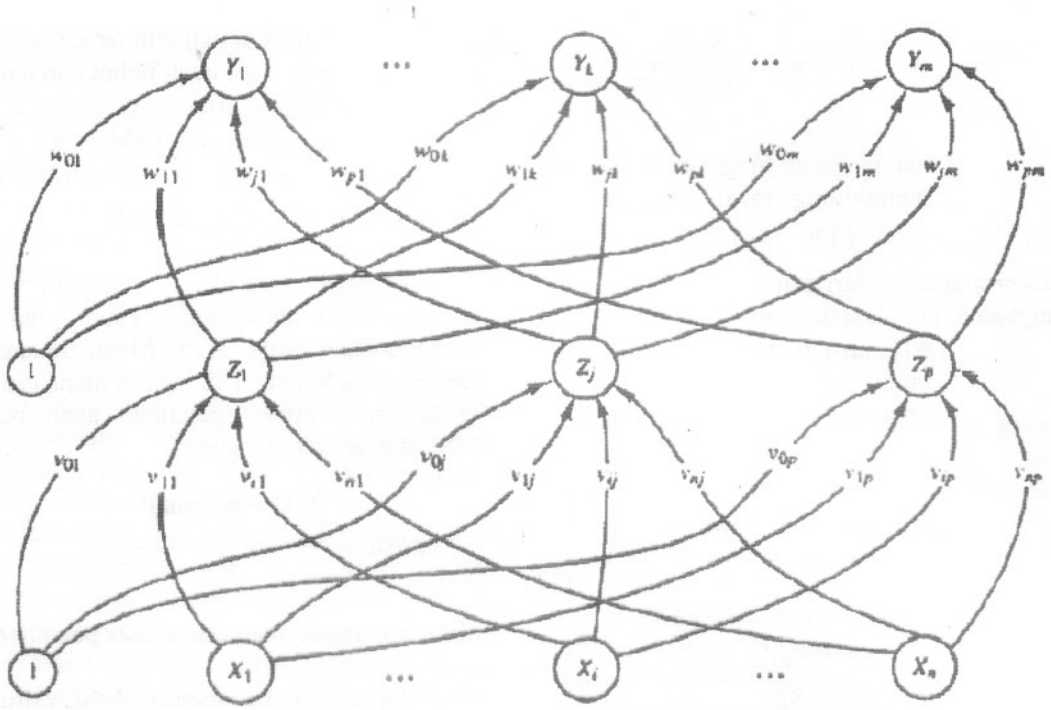
Backpropagation merupakan jaringan *multilayer*. *Backpropagation* termasuk algoritma pembelajaran yang terawasi. Nilai bobot-bobotnya diubah menggunakan *error output* dalam arah mundur (*backward*). Untuk mendapatkan error ini, tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu. Pada perambatan maju, neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi sigmoid.

Arsitektur jaringan propagasi seperti terlihat pada gambar 2. Sebuah jaringan syaraf tiruan lapis banyak dengan satu lapis unit tersembunyi (unit Z) ditunjukkan pada gambar. Unit keluaran (unit Y) dan unit tersembunyi juga memiliki bias. Bias pada unit keluaran Y_K ditunjukkan dengan w_{0K} dan bias pada unit tersembunyi Z_j adalah v_{0j}. Bias berlaku seperti bobot pada hubungan dan memiliki nilai 1.

Algoritma *backpropagation* terdiri dari tiga tahap yaitu *feedforward* untuk input pola pelatihan, perhitungan dan *backpropagation* dari *error* dan perbaikan bobot. Setelah proses pelatihan, aplikasi jaringan yang dipakai hanya perhitungan pada tahap *feedforward*. Ketika proses pelatihan lambat, pelatihan jaringan dapat menghasilkan *output* dengan cepat. Macam variasi dari *backpropagation* telah dibangun untuk menunjang kecepatan dari proses pelatihan [2].

Selama tahap feedforward (maju), setiap unit masukan (X_i) menerima sinyal masukan dan mengirim sinyal ini ke setiap unit tersembunyi $Z_1,\ldots,Z_p$. Setiap unit tersembunyi kemudian menghitung aktifasinya dan mengirim sinyalnya (z_j) ke setiap unit keluaran. Setiap unit keluaran (Y_K) menghitung aktifasinya (y_K) untuk menunjukkan respon jaringan terhadap pola masukan yang diberikan.

Selama pelatihan, setiap unit keluaran membandingkan aktifasi y_K dengan targetnya t_K untuk menentukan galat antara pola masukan dengan unit keluaran tersebut. Setelah didapat galat, faktor δ_K (k=1,...,m) dihitung. δ_K digunakan untuk mendistribusikan galat pada unit keluaran Y_K kembali ke seluruh unit pada lapis sebelumnya (unit tersembunyi yang terhubung dengan Y_K). Kemudian galat ini dipakai untuk mengubah bobot antara keluaran dengan lapisan tersembunyi. Dengan cara yang sama, faktor δj (j = 1,...,p) dihitung untuk setiap unit Z_j. Faktor δj digunakan untuk mengubah bobot antara lapisan tersembunyi dengan lapisan masukan.



Gambar 2 .Jaringan syaraf tiruan Backpropagation dengan satu lapisan tersembunyi

Setelah seluruh faktor δ ditentukan, bobot untuk seluruh lapisan langsung disesuaikan. Penyesuaian bobot w_{jK} (dari unit tersembunyi Z_j ke unit keluaran Y_K) didasarkan pada faktor δ_K dan aktifasi dari unit Z_j, yaitu z_j. Penyesuaian bobot v_{ij} (dari unit masukan X_i ke unit tersembunyi Z_j) adalah didasarkan pada faktor δj dan aktifasi unit masukan x_i.

Fungsi aktifasi yang biasanya dipakai untuk melatih jaringan syaraf tiruan *backpropagation* adalah fungsi sigmoid, baik yang biner maupun bipolar. Berikut algoritma pelatihannya [2] :

Langkah 0. Inisialisasi bobot (tentukan suatu nilai random kecil)

Langkah 1. Selama kondisi berhenti adalah salah, lakukan langkah 2-9 :

Langkah 2. Untuk setiap pasangan pelatihan, lakukan langkah 3-8 :

(*Feedforward*)

Langkah 3. Setiap unit masukan (X_i, i = 1,...n) menerima sinyal masukan x_i dan mengirim sinyal ini ke seluruh unit pada lapisan berikutnya (lapisan tersembunyi).

Langkah 4. Untuk setiap unit tersembunyi (Z_j, j = 1,...,p) jumlahkan sinyal masukan terboboti,

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad (8)$$

terapkan fungsi aktifasi untuk menghitung keluarannya,

$$z_j = f(z\_in_j), \quad (9)$$

dan mengirim sinyal ini ke seluruh unit pada lapisan berikutnya (lapisan keluaran).

Langkah 5. Untuk setiap unit keluaran (Y_k, k =1,...,m), jumlahkan sinyal masukan terbobotinya,

$$y\_in_k = w_{0k} + \sum_{i=1}^{p} z_j w_{jk} \quad (10)$$

dan terapkan fungsi aktifasi untuk menghitung sinyal keluarannya,

$$y_k = f(y\_in_k). \quad (11)$$

Backpropagation dari galat.

Langkah 6. Untuk setiap unit keluaran (Y_k, k =1,...,m) menerima sebuah pola target yang bersesuaian dengan pola masukan, hitung galatnya,

$$\delta_k = (t_k - y_k) f'(y\_in_k), \quad (12)$$

hitung koreksi bobotnya,

$$\Delta w_{jk} = \alpha \delta_k z_j, \quad (13)$$

hitung koreksi biasnya,

$$\Delta w_{0k} = \alpha \delta_k, \quad (14)$$

dan kirim δ_k ke unit pada lapisan sebelumnya.

Langkah 7. Untuk setiap unit tersembunyi (Z_j, j = 1,...,p) jumlahkan masukan deltanya,

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad (15)$$

kemudian dikalikan dengan turunan fungsi aktifasinya untuk menghitung galatnya,

$$\delta_j = \delta\_in_j f'(z\_in_j), \quad (16)$$

hitung koreksi bobotnya ,

$$\Delta v_{ij} = \alpha \delta_j x_i, \quad (17)$$

hitung koreksi biasnya,

$$\Delta v_{0j} = \alpha \delta_j. \quad (18)$$

Ubah bobot dan bias :

Langkah 8. Untuk setiap unit keluaran (Y_k, k =1,...,m) ubah bias dan bobotnya (j = 1,...,p) :

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk}. \quad (19)$$

Untuk setiap unit tersembunyi (Z_j, j = 1,...,p) ubah bobot dan biasnya (i = 1,...n):

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij}. \quad (20)$$

Langkah 9. Tes kondisi berhenti.

Alternatif kondisi berhenti selain menggunakan maksimum epoh juga dapat menggunakan target *error* Mean Square Error (MSE). Jika target MSE sudah mencapai target error, maka proses pelatihan akan berhenti. MSE dinyatakan dengan

$$MSE = \frac{\sum_{i=1}^{n} (t - output)^2}{n} \quad (21)$$

dimana *n* adalah banyaknya data pelatihan.

Langkah-langkah mencari MSE yaitu :

1. Hitung selisih antara nilai target dengan nilai *output* dari pelatihan.
2. Kuadratkan setiap selisih tersebut.
3. Jumlahkan semua kuadrat selisih dari tiap-tiap data pelatihan dalam satu epoh.
4. Bagi hasil penjumlahan tersebut dengan banyaknya data pelatihan.

Setelah proses pelatihan, jaringan syaraf *backpropagation* diaplikasikan dengan menggunakan fase *feedforward* dari algoritma pelatihan. Algoritma untuk pengujian adalah sebagai berikut :

1. Operasi pada *hidden layer* : menjumlahkan bobot akhir *input* beserta bobot akhir bias ke *hidden*.

   $$z\_in_j = v_{0j} + \sum x_i v_{ij} \quad (22)$$

   Fungsi aktifasi untuk menghitung sinyal *output*nya :

   $$z_j = f(z\_in_j) \quad (23)$$

2. Operasi pada *output layer* : menjumlahkan bobot akhir *hidden* ke *output* beserta bobot akhir bias ke *output*.

   $$y\_in_k = w_{0k} + \sum_{i=1}^{n} z_i w_{jk} \quad (24)$$

   Gunakan fungsi aktifasi untuk menghitung sinyal *output*nya :

$$y_k = f(y\_in_k) \quad (25)$$

3. Analisis dan Perancangan Sistem

Analisis dan Kebutuhan Sistem

Citra yang direpresentasikan dalam bentuk matrik memiliki dimensi yang besar dengan baris dan kolom menunjukkan sebuah titik pada citra serta kesesuaian nilai elemen matrik mengidentifikasikan level pada titik tersebut. Proses komputasi dimensi matrik yang besar membutuhkan waktu proses yang lama sehingga diperlukan reduksi dimensi matrik untuk meminimalisir waktu proses.

Identifikasi dan pengenalan wajah didasari pada pengenalan pola dengan pendekatan jaringan syaraf. Metode jaringan syaraf yang dipergunakan adalah *backpropagation*.

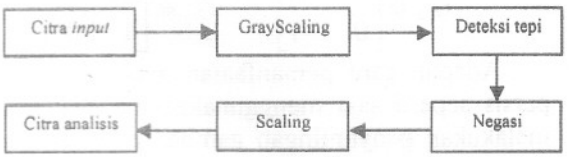
Proses identifikasi wajah dilakukan dengan membandingkan citra input dengan citra yang telah dilatih oleh sistem. Secara garis besar, penelitian ini terdiri dari beberapa tahap yaitu :

1. Pengambilan citra digital.
2. Proses pengolahan citra yang meliputi *grayscaling*, deteksi tepi berbasis gradien dengan operator Roberts, proses operasi negasi, dan *scaling*.
3. Proses membuat data *input* yang digunakan dalam proses jaringan syaraf tiruan.
4. Proses pelatihan dan pengujian.

Pengolahan Data

Data yang digunakan dalam penelitian ini adalah sekumpulan citra untuk basis data (*database data set*) dan sekumpulan citra untuk pengujian (*testing data set*) yang diperoleh dari internet dengan ekspresi yang berbeda-beda. Citra yang digunakan untuk basis data tidak akan digunakan nantinya pada saat pengujian.

Citra wajah yang digunakan berukuran 92x112 pixel. Citra tersebut mengalami proses konversi ke *grayscaling* jika merupakan citra *true color*, selanjutnya mengalami proses deteksi tepi berbasis gradien dengan operator Roberts, proses operasi negasi dan *scaling* sehingga menghasilkan citra baru dengan ukuran 30x36 pixel yang merupakan citra yang akan digunakan.



Gambar 4. Proses pengolahan data

Citra analisis merupakan citra *grayscale* dengan skala 8 bit yang memiliki intensitas warna berkisar antara 0 sebagai nilai minimum sampai 255 yang merupakan nilai maksimum.

Citra analisis yang memiliki ukuran 30x36 pixel kemudian dikonversi kedalam bentuk vektor 30x36 = 1080, untuk masing-masing citra. Untuk setiap pixel dengan nilai intensitas warna lebih kecil sama dengan 237 diberi nilai 1 sedangkan untuk nilai intensitas warna lebih besar 237 diberi nilai 0, sehingga akan diperoleh barisan nilai yang terdiri 0 dan 1 (barisan bilangan biner) sepanjang 1080.

Untuk setiap 24 bit data citra dari barisan bilangan 0 dan 1 sepanjang 1080 selanjutnya akan dikonversi ke dalam bentuk desimal sehingga menghasilkan barisan bilangan desimal sebanyak 45. Dari 45 bilangan desimal tersebut kemudian dibuat suatu pola *input* dengan aturan sebagai berikut :

```
if nilaidesimal<=3355443 then
   nilaiinput:=0
else if nilaidesimal <= 6710886 then
   nilai input:=0.25
else if nilaidesimal <=10066329 then
   nilaiinput:=0.5
else if nilaidesimal <=13421772 then
     nilaiinput:=0.75
else  nilaiinput:=1;
```

Dari proses di atas akan menghasilkan 45 nilai input yang nantinya digunakan pada proses pelatihan dan pengujian citra wajah.

Deteksi Tepi

Deteksi tepi (*edge detection*) dengan operator Roberts merupakan salah satu pengembangan dari teknik deteksi tepi yang

telah dibahas dengan memiliki *mask* berukuran 2x2.

$$G_x \approx \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad G_y \approx \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Adapun cara pemanfaatan *mask* ini sama persis seperti saat menggunakan matrik untuk melakukan penyuntingan gambar yaitu dengan memasukkan titik-titik disekitar titik yang diperiksa kedalam matrik 2x2. Kemudian dari kombinasi antara kedua hasil konvolusi *mask* tadi dilakukan dengan pengambilan nilai maksimum.

Sebagai contoh adalah sebuah citra berukuran 6x5 pixel, dengan nilai tertentu disetiap pixelnya akan dilakukan deteksi tepi, yaitu sebagai berikut:

- Nilai dari citra di nyatakan dalam bentuk matrik 6x5

w →

h ↓

	0	1	2	3	4	5
0	7	255	255	255	255	255
1	7	255	255	7	7	255
2	7	8	7	7	255	255
3	7	7	255	255	255	255
4	7	8	7	255	255	255

- *Maks* yang bersesuaian adalah

Mask1 = v ↓ $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ u →

Mask2 = v ↓ $\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ u →

- Prosesnya yaitu:

```
Untuk x = 1 sampai w-2
  Untuk y = 1 sampai h-2
  Mulai
    Konvolusi1 = 0
    Untuk u = 0 sampai 1
      Untuk v = 0 sampai 1
        Konvolusi1   =   konvolusi1   +
          mask1[u,v]                  *
          nilai_citra_lama[x-u,y-v]
    Konvolusi2 = 0
    Untuk u = 0 sampai 1
      Untuk v = 0 sampai 1
        Konvolusi2   =   konvolusi2   +
          mask2[u,v]                  *
          nilai_citra_lama[x-u,y-v]
    Nilai_citra_baru[x,y]   =   255   -
      (round(abs(konvolusi1)
      +abs(konvolusi2)))
  Selesai
```

- Matrik nilai dari hasil deteksi tepi yaitu:

	0	1	2	3	4	5
0	7	255	255	255	255	255
1	7	15	255	7	15	255
2	7	6	16	7	7	255
3	7	254	8	15	7	255
4	7	8	7	255	255	255

Pengurangan 255 terhadap nilai hasil penjumlahan 2 konvolusi dari citra dimaksudkan untuk menegasikan, karena hasil konvolusi biasanya berada disekitar nilai 0 (nilai kecil), sehingga jika ditampilkan, citra tepi kebanyakan berwarna gelap (hitam). Untuk citra *true color* dilakukan konversi terlebih dahulu ke bentuk *grayscaling* sebelum dilakukan deteksi tepi.

Perancangan Jaringan Syaraf Tiruan

Arsitektur jaringan syaraf yang digunakan disini terdiri dari 3 layer, yaitu 1 layer *input*, 1 *hidden layer* dan 1 *layer output*. Jumlah *neuron input* kejaringan syaraf tiruan dengan algoritma *backpropagation* adalah 45 *neuron* yang diperoleh dari pengolahan data sebagai mana yang telah dibahas sebelumnya. Jumlah *neuron hidden layer* yaitu sebanyak *n* yang dimasukkan oleh *user*. Untuk jumlah *neuron output layer* sebanyak 2 buah *neuron* dengan fungsi aktivasi yang digunakan adalah fungsi sigmoid biner dengan nilai range 0 sampai 1.

Target pengujian (target *output*) yang merupakan hasil keluaran dari *neuron output layer* adalah sebanyak 4 jenis kelas pola wajah. Untuk cita wajah pelatihan dengan identitas ke-1 memiliki taget 00, citra wajah pelatihan dengan identitas ke-2 memiliki target 01, citra wajah pelatihan dengan identitas ke-3 memiliki target 10, dan cita wajah pelatihan dengan identitas ke-4 memiliki target 11.

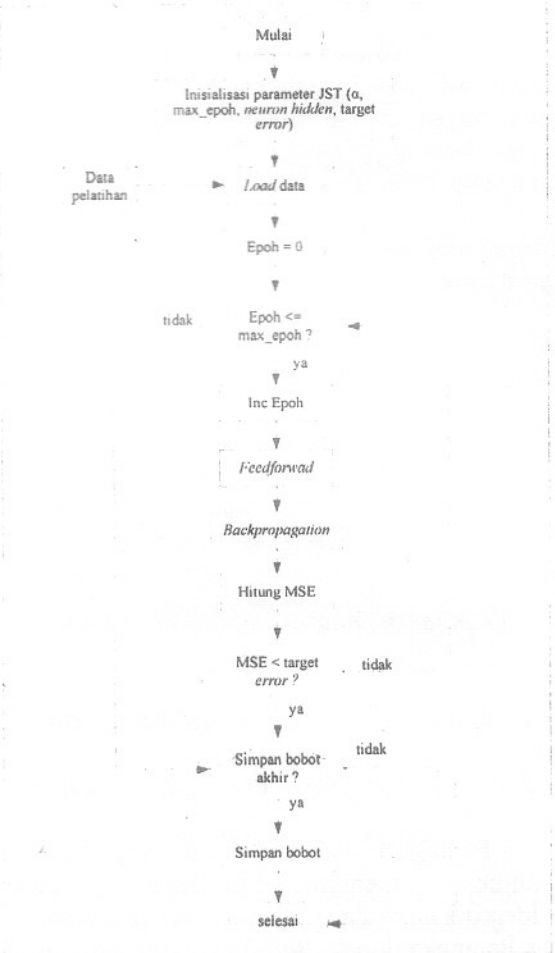
Proses Pelatihan

Proses pelatihan diawali dengan proses inisialisasi yang diikuti dengan tahapan *feedforwad* yaitu menjumlahkan sinyal-sinyal *input* terbobot pada *hidden layer* dan *output layer*. Sinyal-sinyal tersebut diaktivasi dengan fungsi sigmoid biner.

Tahap selanjutnya adalah tahap *backpropagation* yang dimulai dengan menghitung informasi *error*. Informasi *error* diperoleh dari selisih antara nilai target yang telah ditentukan dengan nilai keluaran dari *output layer*. Informasi *error* tersebut digunakan untuk mengoreksi bobot pada unit *output* dan *hidden*. Koreksi bias dilakukan pula pada unit *output* dan unit *hidden*. Tiap-tiap unit *output* dan unit *hidden* memperbaiki bobot dan biasnya. Langkah tersebut diatas dikerjakan berulang-ulang selama kurang dari maksimum epoh atau kuadrat *error* kurang dari target *error*.

Diagram alir proses pelatihan dapat dilihat pada gambar 5 di samping.

Untuk *Mean Square Error* (MSE) dari proses pelatihan diperoleh dari penjumlahan kuadrat *error* setiap pelatihan selama satu kali epoh dibagi dengan banyaknya data yang dilatihkan. Iterasi epoh yang semakin banyak cenderung akan menghasilkan nilai *error* yang semakin kecil. Tetapi pelatihan yang optimal dipengaruhi oleh banyak faktor seperti penentuan *learning rate*, banyaknya *hidden layer* serta arsitektur jaringannya sendiri.



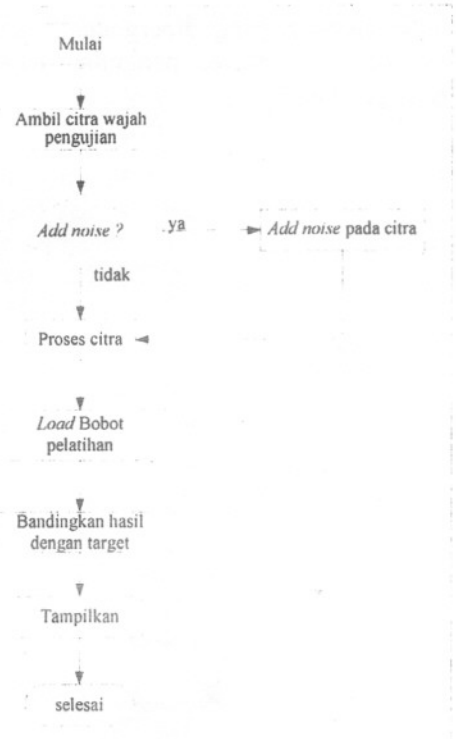
Gambar 5. Diagram alir proses pelatihan

Proses Pengujian

Proses pengujian diawali dengan mengambil citra wajah yang akan diujikan. Citra tersebut dapat diberikan nilai *noise* yang selanjutnya mengalami proses citra.

Pengujian dilakukan dengan menghitung penjumlahan bobot akhir pada *hidden layer* dan *output layer* dengan citra wajah yang akan diujikan. Tahap pengujian ini seperti tahap pada *feedforwad* yang menghasilkan nilai *output error*. Nilai *output* yang telah diaktivasi ini dibandingkan dengan nilai ambang (*treshold*). Nilai ambang yang dipergunakan adalah 0,5. Diagram alir proses pengujian dapat dilihat pada gambar 6.

Setelah diperoleh target *output*, kemudian dilakukan proses perhitungan nilai tepi antara citra yang diujikan dengan citra-citra dengan target *output* yang sesuai. Persentase nilai tepi terbesar antara citra yang diujikan dengan citra hasil pengujian akan ditampilkan sebagai citra hasil pengujian yang paling mendekati dengan citra yang diujikan.



Gambar 6. Diagram alir proses pengujian

4. Implementasi dan Pengujian Sistem

Implementasi Perangkat Lunak

Perangkat lunak (*software*) yang digunakan untuk mengimplementasikan Sistem Identifikasi Wajah ini adalah sebagai berikut:

1. Perangkat lunak, *Borland Delphi 7* digunakan untuk membangun Aplikasi.
2. Database, *MySQL 5.0* digunakan untuk menyimpan dan mengelola basis data.
3. Sistem Operasi Windows XP.

Pengujian Sistem

Proses pengujian menggunakan citra standart yang didapat dari The Database of Faces, AT&T Laboratories Cambridge [15]. Data pelatihan yang digunakan sebanyak 4, dengan 5 buah citra wajah pada masing-masing data pelatihan. Pengujian menggunakan 4x5 buah citra yang berbeda dengan citra yang dilatihkan.

Hasil pengujian dengan inisialisasi parameter *learning rate* 0.5, target *error* 0.01, epoh 1000, *neuron hidden layer* 10 dapat dilihat pada tabel 1 berikut ini:

Tabel 1. Pengujian Citra Tanpa Flip

No	Citra uji	Citra hasil pengujian	Persentase nilai tepi	Dikenali benar
1	6.bmp	4.bmp	81.30%	Ya
2	7.bmp	4.bmp	77.78%	Ya
3	8.bmp	2.bmp	79.63%	Ya
4	9.bmp	4.bmp	74.44%	Ya
5	10.bmp	183.bmp	69.35%	Tidak
6	106.bmp	102.bmp	69.44%	Ya
7	107.bmp	133.bmp	63.70%	Tidak
8	108.bmp	102.bmp	75.00%	Ya
9	109.bmp	105.bmp	67.50%	Ya
10	1010.bmp	102.bmp	66.02%	Ya
11	136.bmp	132.bmp	71.57%	Ya
12	137.bmp	131.bmp	72.31%	Ya
13	138.bmp	131.bmp	72.78%	Ya
14	139.bmp	135.bmp	69.26%	Ya
15	1310.bmp	135.bmp	77.50%	Ya
16	186.bmp	183.bmp	69.54%	Ya
17	187.bmp	181.bmp	72.69%	Ya
18	188.bmp	183.bmp	71.30%	Ya
19	189.bmp	182.bmp	73.15%	Ya
20	1810.bmp	185.bmp	71.02%	Ya

Dari hasil pengujian yang dilakukan, keberhasilan yang diperoleh mencapai nilai 90% dengan persentase kemiripan nilai tepi lebih dari 50%.

Untuk pengujian lebih lanjut digunakan perlakuan flip (pencerminan) terhadap citra uji. Proses *flip* terhadap citra dilakukan di luar sistem. Pengujian ini juga menggunakan parameter yang sama dengan sebelumnya yaitu *learning rate* 0.5, target *error* 0.01, epoh 1000, *neuron hidden layer* 10. Hasil pengujian dapat dilihat pada tabel 2 di bawah ini.

Dari hasil pengujian terhadap citra yang di-*flip* menunjukkan bahwa keberhasilan yang diperoleh adalah sebesar 50%. Penurunan keberhasilan ini disebabkan perubahan pola tepian yang dibentuk oleh citra dari semula tidak di-*flip* yang kemudian di-*flip* mengakibatkan kesalahan pada *input* bagi

jaringan syaraf. Sehingga dapat mengakibatkan kesalahan dalam identifikasi citra.

Tabel 2. Pengujian Citra Dengan Fflip

No	Citra uji	Citra hasil pengujian	Persentase nilai tepi	Dikenali benar
1	6.bmp	103.bmp	69.26%	Tidak
2	7.bmp	5.bmp	76.67%	Ya
3	8.bmp	133.bmp	68.70%	Tidak
4	9.bmp	2.bmp	72.22%	Ya
5	10.bmp	5.bmp	73.33%	Ya
6	106.bmp	101.bmp	70.74%	Ya
7	107.bmp	101.bmp	67.50%	Ya
8	108.bmp	181.bmp	60.46%	Tidak
9	109.bmp	103.bmp	69.81%	Ya
10	1010.bmp	3.bmp	68.98%	Tidak
11	136.bmp	185.bmp	65.74%	Tidak
12	137.bmp	103.bmp	62.22%	Tidak
13	138.bmp	102.bmp	61.30%	Tidak
14	139.bmp	132.bmp	68.52%	Ya
15	1310.bmp	4.bmp	65.56%	Tidak
16	186.bmp	182.bmp	71.39%	Ya
17	187.bmp	185.bmp	69.26%	Ya
18	188.bmp	103.bmp	64.72%	Tidak
19	189.bmp	183.bmp	70.37%	Ya
20	1810.bmp	5.bmp	67.96%	Tidak

5. Penutup

Kesimpulan

Dari penelitian yang dilakukan dapat ditarik beberapa kesimpulan diantaranya sebagai berikut:

1. Ukuran citra yang dapat dipergunakan dalam proses identifikasi adalah citra dengan ukuran 92x112 pixel dengan dengan tipe bmp (Bitmap).
2. Inisialisasi parameter mempengaruhi hasil dari proses pelatihan yang nantinya mempengaruhi juga hasil pengujian.
3. Dari hasil pengujian yang dilakukan dengan Inisialisasi parameter *learning rate* 0.5, target *error* 0.01, epoh 1000, *neuron hidden layer* 10 dan *learning rate* 0.5, target *error* 0.01, epoh 500, *neuron hidden layer* 10 keberhasilan yang di peroleh adalah sebesar 90%.
4. Citra yang di-*flip* dapat mengakibatkan kesalahan dalam identifikasi citra. Hal ini disebabkan terjadi perubahan pola tepian yang dibentuk oleh citra.

Saran

Dalam pengembangan sistem identifikasi dengan deteksi tepi masih banyak bagian yang masih perlu dilakukan penelitian lebih lanjut untuk mendapatkan hasil yang lebih maksimal diantaranya adalah:

1. Penggunaan deteksi tepi yang lain seperti operator Prewitt, Sobel, Canny atau yang lainnya, dalam mengolah citra.
2. Penggunaan metode lain dalam membentuk nilai *input*-an kejaringan syaraf tiruan misalnya dengan *cell* matriks.
3. Penggunaan jaringan syaraf *backpropagation* dengan bentuk variasi yang lain ataupun jaringan syraraf tiruan dengan metode yang lain.
4. Pengembangan sistem identifikasi citra wajah yang mengarah ke pola 3 dimensi.

Daftar Pustaka

[1] Achmad, B. dan Firdausy, K., 2005, Teknik Pengolahan Citra Digital Menggunakan Delphi, Ardi Publishing, Yogyakarta.

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

[3] Gonzales, R. C. dan R. E. Woods, 1992, Digital Image Processing, Addison Wesley, Massachusetts.

[4] Kusumadewi, S., 2003, Artificial Intelligence (Teknik dan Aplikasinya), Graha Ilmu, Yogyakarta.

[5] Low, Andrian, 1991, Introduction Computer Vision and Image Processing, Mc Graw Hill Book Company, London.

[6] Murni, A., dan Chahyanti, D., 2004, Edge Detection, Fakultas Ilmu Komputer Universitas Indonesia.

[7] Schalkoff, R., 1992, Pattern Recognition Statistical, Structural And Neural Approaches, Jhon Wiley & Sons, Inc., New York.

[8] Campilho, A., 2004, Edge Detection, http://paginas.fe.up.pt/~campilho/APSIM/NOTES/2004/EdgeDetection.pdf (7 Juli 2006).

[9] The Database of Faces, AT&T Laboratories Cambridge, http://www.cl.cam.ac.uk/Research/DTG/attarchive/pub/data/att_faces.zip (7 Juli 2006).