BAB III
ANALISA SISTEM

3.1 Teknik Pengumpulan Data

Data yang digunakan berupa citra buah pepaya (Carica Papaya) mentah, setengah matang, matang. Gambar diambil dengan kamera handphone xiaomi dengan ukuran piksel 8,0.

3.1.1 Bahan Penelitian

Data set yang digunakan pada penelitian ini bersumber pada hasil pemotretan citra sebanyak 40 foto buah pepaya yang diambil dari toko buah di Kabupaten Mojokerto dengan tingkat kematangan berbeda-beda seperti pada Gambar 3.1.

(a) (b) (c)

Gambar 3.1 Tingkat Kematangan Pepaya

Pada gambar 3.1 menjelaskan bahwa terdapat 3 kematangan buah pepaya yaitu: (a) mentah, (b) setengah matang dan (c) matang. Pada penelitian ini dibatasi hanya menggunakan 30 data citra buah pepaya sebagai data training dan 15 data citra buah pepaya sebagai data testing. Rincian data untuk proses training dan proses testing adalah sebagai berikut:

1. Data training
    a. 10 data citra buah pepaya mentah
    b. 10 data citra buah pepaya setengah matang
    c. 10 data citra buah pepaya matang

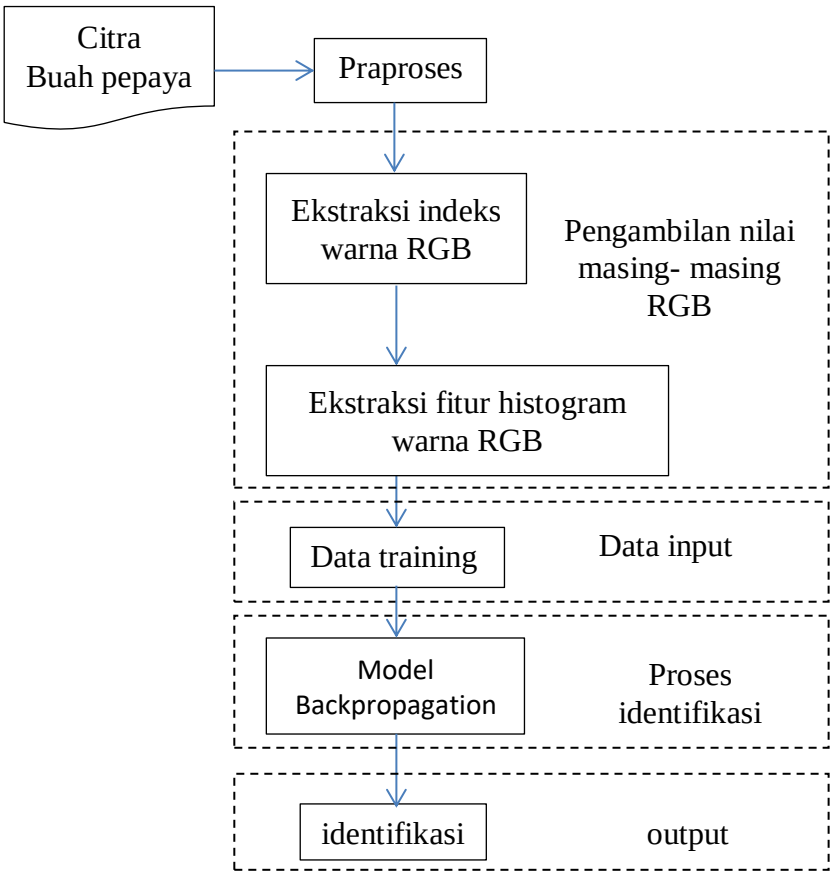
2. Data testing
    a. 5 data citra buah pepaya mentah
    b. 5 data citra buah pepaya setengah matang
    c. 5 data citra buah pepaya matang

3.2 Teknik Analisis Data Metode Backpropagation

Analisis data bertujuan untuk mengklasifikasikan tingkat kematangan buah pepaya dengan menggunakan model backpropagation. Tahapan yang dilakukan yaitu:

1. Studi Literatur
2. Pra Proses Meliputi : input buah pepaya, citra hasil input akan mengalami image processing yaitu proses histogram warna.
3. Proses Identifikasi Kematangan Buah Pepaya
    1. Arsitektur Backpropagation
    2. Rancangan pembelajaran (training) backpropagation : Citra – histogram warna – pola – training – bobot.
    3. Rancangan proses identifikasi backpropagtion : Citra – histogram warna – pola – indentifikasi (dari bobot).

Secara singkat klasifikasi kematangan buah pepaya dengan menggunakan model backpropagation jika digambarkan dalam diagram sebagai berikut:



Gambar 3.2 tahap penelitian backpropagation

Gambar diatas menjelaskan tahapan-tahapan proses penelitian yang pertama mulai dari pengambilan bahan penelitian, kemudian proses yang kedua akuisisi citra RGB, kemudian yang ketiga proses training data input dan yang terakhir rancangan proses dalam tahapan deteksi tingkat kematangan buah pepaya. Pada Gambar 3.2 citra akan diproses satu persatu berdasarkan tahapan yang dilalui mulai dari input sampai dengan output. Dan hasil berupa citra pepaya yang dideteksi berdasarkan tingkat kematangan.

3.3 Perhitungan Backpropagation

Metode Backpropagation yang dibangun pada tugas akhir ini menggunakan fungsi aktivasi *tansig* pada lapisan tersembunyi dan fungsi aktivasi *traingdm* pada lapisan output. Secara matematis model Backpropagation dapat dirumuskan sebagai berikut:

Penjumlahan sinyal-sinyal x_i terbobot z_in_j dan sinyal output pada lapisan input menuju ke lapisan tersembunyi z_j adalah

$$\begin{aligned} z\_in_j &= v_{oj} + (v_{1j}x_1 + v_{2j}x_2 + \cdots + v_{nj}x_n \\ &= v_{oj} + \sum_{i=1}^{n} v_{ij}x_i \end{aligned} \tag{3.1}$$

$$z_j = f(z\_in_j) = \frac{1-e^{-z\_in_j}}{1+e^{-z\_in_j}} = \frac{1-e^{-(v_{oj}+\sum_{i=1}^{n} v_{ij}x_i}}{1+e^{-(v_{oj}+\sum_{i=1}^{n} v_{ij}x_i}} \tag{3.2}$$

Penjumlahan sinyal-sinyal output z_j terbobot y_in dan sinyal output pada lapisan tersembunyi menuju ke lapisan output (y) adalah

$$\begin{aligned} y\_in &= w_0 + w_1z_1 + w_2z_2 + \cdots + w_pz_p \\ &= w_0 + \sum_{j=1}^{p} w_jz_j. \\ &= w_0 + \sum_{j=1}^{p} w_j \left(\frac{1-e^{-(v_{oj}+\sum_{i=1}^{n} v_{ij}x_i}}{1+e^{-(v_{oj}+\sum_{i=1}^{n} v_{ij}x_i}} \right) \end{aligned} \tag{3.3}$$

$$\begin{aligned} y &= f(y\_in) + \varepsilon \\ &= y\_in + \varepsilon \\ &= w_0 + \sum_{j=1}^{p} w_j \left(\frac{1-e^{-(v_{oj}+\sum_{i=1}^{n} v_{ij}x_i}}{1+e^{-(v_{oj}+\sum_{i=1}^{n} v_{ij}x_i}} \right) + \varepsilon \end{aligned} \tag{3.4}$$

Karena lapisan output merupakan lapisan terakhir, maka sinyal output yang menuju ke lapisan output adalah nilai output jaringan yang diinginkan. Oleh karena itu, persamaan (3.4) merupakan rumus umum metode backpropagation, dengan:

y = nilai output

x_i = variabel input, dengan i = 1,2,...,n,

v_{ij} = bobot-bobot yang menghubungkan neuron input ke-i menuju neuron ke-j pada lapisan tersembunyi.

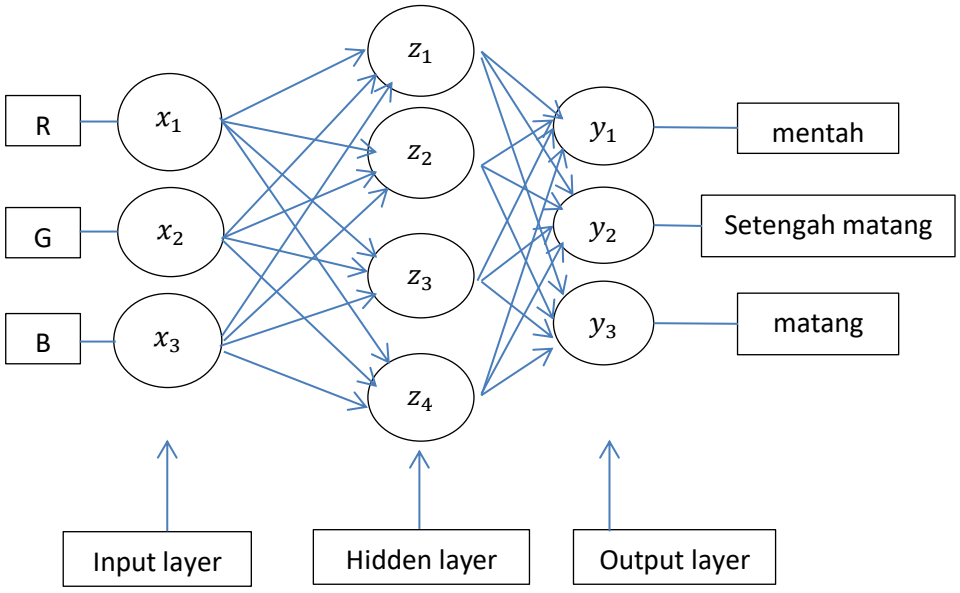
w_j = bobot dari neuron ke-j pada lapisan tersembunyi yang menuju lapisan output, dengan j = 1,2,...,p,

v_{oj} = bobot bias yang menuju neuron ke-j pada lapisan tersembunyi, dengan j = 1,2,...,p,

w_0 = bobot bias yang menuju neuron pada lapisan output,

ε = error.

3.4 Arsitektur Backpropagation



Gambar 3.3. Arsitektur Backpropagation

Perancangan arsitektur backpropagation ini menggunakan 3 unit input, 4 node hidden layer dan 3 unit output.

Keterangan :

x_1, x_2, x_3 : nilai input variabel RGB

z_1, z_2, z_3, z_4 : neuron hidden layer 1

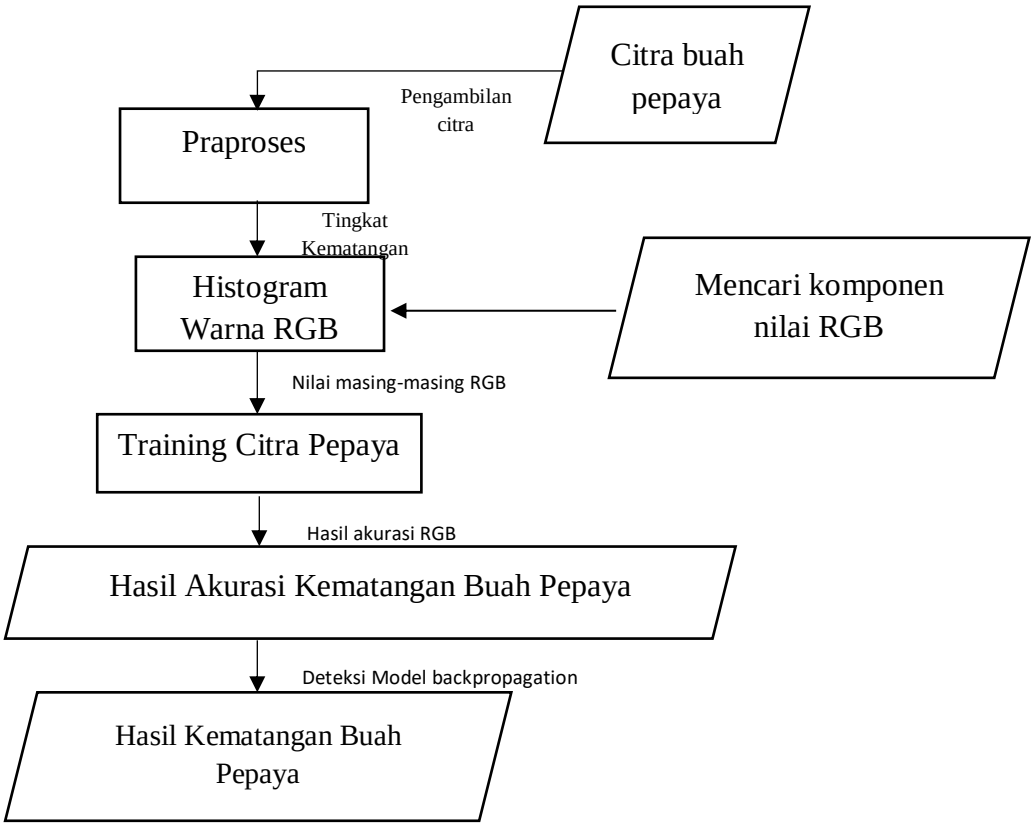
y_1, y_2, y_3 : nilai target, bernilai -1 jika mentah, 0 jika setengah matang dan 1 jika matang.

Pada gambar gambar 3.3 diperlihatkan jaringan dengan 3 buah unit masukkan (x_1, x_2, x_3), sebuah hidden layer yang terdiri atas 4 buah neuron (z_1, z_2, z_3, z_4) dan 3 unit keluaran y_1, y_2, y_3).

3.5 Algoritma Sistem

Setelah gambar diambil dari komputer atau memasukkan gambar pada aplikasi, sistem melakukan langkah-langkah berikut untuk memprediksi tingkat kematangan pada buah pepaya seperti pada Gambar 3.4.

3.5.1 Algoritma training dan Uji sistem backpropagation



Gambar 3.4. Algoritma training dan Uji Backpropagation Sistem Prediksi Kematangan Buah Pepaya

Rancangan algoritma pada Gambar 3.4 yang digunakan untuk sistem prediksi kematangan buah pepaya terdapat 3 fungsi utama pada aplikasi, yaitu fungsi deteksi histogram warna RGB adalah untuk menerapkan segmentasi warna tingkat kematangan buah pepaya dan mencari nilai masing-masing RGB, kemudian yang kedua fungsi training citra pepaya adalah memelatih citra buah pepaya dengan tingkat kematangan tertentu untuk memperoleh hasil akurasi. Kemudian yang ketiga fungsi deteksi citra pepaya adalah mendeteksi buah pepaya dengan tingkat tertentu.

3.5.2 Akuisisi Citra

Dalam proses ini dibutuhkan suatu alat pengambilan citra dengan cara pemotretan citra dengan latar belakang yang terang atau warna putih. Alat yang digunakan untuk pengambilan citra dengan menggunakan kamera Ponsel dengan ukuran 5 Megapixel dengan menggunakan automatic flash.

3.5.3 Rancangan Proses

Pada sub bab ini akan dijelaskan tahapan-tahapan proses deteksi buah pepaya berdasarkan tingkat kematangan mulai proses *input* sampai *output*. Tahapan-tahapan ini antara lain:

3.5.4 Citra Buah Pepaya dan Praproses

Citra yang diinputkan berupa foto buah pepaya yang diambil dari pasar di Kabupaten Mojokerto.

Praproses adalah proses pengambilan citra pada sistem database komputer dan memasukkan citra pada aplikasi. Didalam praproses ini aplikasi akan mendeteksi model warna RGB secara otomatis pada citra.

3.5.5 Histogram Citra RGB

Pada proses ini citra akan otomatis terdeteksi nilai masing-masing dari komponen warna RGB sesuai kematangan buah pepaya. Dari hasil deteksi nilai warna RGB akan diketahui yaitu histogram nilai max, min dan mean RGB. Berikut rumus untuk menghitung komponen nilai masing-masing RGB.

$$R = \frac{R}{R+G+B} \tag{3.5}$$

$$G = \frac{G}{R+G+B} \tag{3.6}$$

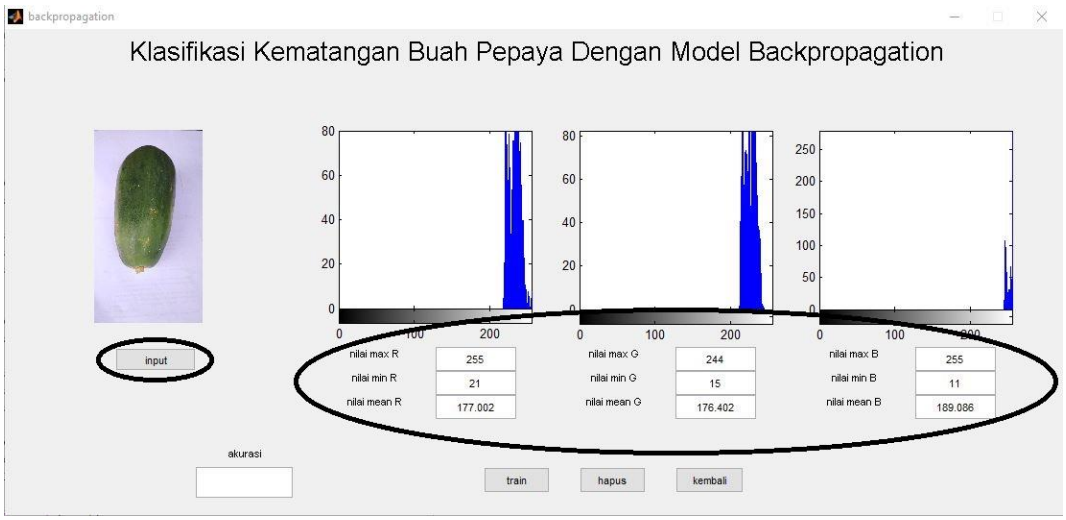
$$B = \frac{B}{R+G+B} \tag{3.7}$$

3.5.6 Training Citra Pepaya

Pada proses ini citra yang sudah diketahui masing-masing komponen nilai dari RGB akan ditraining. Proses training berguna untuk melatih sistem dengan memasukkan data-data inputan ke dalam sistem Neural Network kemudian data tersebut diolah dengan menggunakan metode backpropagation. Proses training data citra buah pepaya bertujuan untuk mendapatkan nilai bobot dan nilai akurasi dari masing-masing kematangan buah pepaya.

3.6 Perancangan Ekstraksi Fitur

Perancangan ekstraksi fitur di Tugas Akhir ini adalah untuk menentukan nilai dari masing-masing komponen RGB. Berikut perancangan program ekstraksi fitur RGB.



Gambar 3.5 Perancangan histogram

```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
[nama_file1, nama_path1]=uigetfile(...
{'*.bmp;*.jpg','File citra(*.bmp,*.jpg)';
'*.bmp','File Bitmap(*.bmp)';...
'*.jpg','File Jpeg(*.jpg)';
'*.*','Semua File(*.*)'});

if~isequal(nama_file1,0);
    handles.data1=imread(fullfile(nama_path1,nama_file1));
    guidata(hObject, handles);
    handles.current_data1=handles.data1;
    axes(handles.axes4);
    imhist(max(handles.current_data1(:,:,3)));
```

```
    axes(handles.axes3);
    imhist(max(handles.current_data1(:,:,2)));
    axes(handles.axes2);
    imhist(max(handles.current_data1(:,:,1)));
    axes(handles.axes1);
    imshow(handles.current_data1);
else
    return
end

red=max(max(handles.current_data1(:,:,1)));
green=max(max(handles.current_data1(:,:,2)));
blue=max(max(handles.current_data1(:,:,3)));

set(handles.edit1,'string',red);
set(handles.edit3,'string',green);
set(handles.edit5,'string',blue);

red=min(min(handles.current_data1(:,:,1)));
green=min(min(handles.current_data1(:,:,2)));
blue=min(min(handles.current_data1(:,:,3)));

set(handles.edit2,'string',red);
set(handles.edit4,'string',green);
set(handles.edit6,'string',blue);

red=mean(mean(handles.current_data1(:,:,1)));
green=mean(mean(handles.current_data1(:,:,2)));
blue=mean(mean(handles.current_data1(:,:,3)));

set(handles.edit9,'string',red);
set(handles.edit10,'string',green);
set(handles.edit11,'string',blue);
```

3.7 Perancangan Arsitektur Jaringan Syaraf Tiruan

Jaringan yang digunakan untuk mendeteksi tingkat kematangan buah pepaya adalah jaringan syaraf tiruan backpropagation dengan langkah pembelajaran feedforward. Jaringan ini memiliki beberapa lapisan, yaitu lapisan masukan, lapisan keluaran dan beberapa lapisan tersembunyi. Lapisan tersembunyi tersebut membantu jaringan untuk dapat mengenali lebih banyak pola masukan dibandingkan dengan jaringan yang tidak memiliki lapisan tersembunyi.

Parameter-parameter dalam pembentukan jaringan backpropagation adalah sebagai berikut.

```
network1 =

    Neural Network object:

    architecture:

         numInputs: 1
         numLayers: 2
       biasConnect: [1; 1]
     inputConnect: [1; 0]
     layerConnect: [0 0; 1 0]
    outputConnect: [0 1]

        numOutputs: 1   (read-only)
    numInputDelays: 0   (read-only)
    numLayerDelays: 0   (read-only)

    subobject structures:

            inputs: {1x1 cell} of inputs
            layers: {2x1 cell} of layers
           outputs: {1x2 cell} containing 1 output
            biases: {2x1 cell} containing 2 biases
      inputWeights: {2x1 cell} containing 1 input weight
      layerWeights: {2x2 cell} containing 1 layer weight

    functions:

          adaptFcn: 'trains'
         divideFcn: 'dividerand'
       gradientFcn: 'gdefaults'
           initFcn: 'initlay'
        performFcn: 'mse'
          plotFcns: {'plotperform','plottrainstate','plotregression'}
          trainFcn: 'trainlm'

    parameters:

        adaptParam: .passes
       divideParam: .trainRatio, .valRatio, .testRatio
     gradientParam: (none)
         initParam: (none)
      performParam: (none)
        trainParam: .show, .showWindow, .showCommandLine, .epochs,
                    .time, .goal, .max_fail, .mem_reduc,
                    .min_grad, .mu, .mu_dec, .mu_inc,
                    .mu_max

    weight and bias values:

                IW: {2x1 cell} containing 1 input weight matrix
                LW: {2x2 cell} containing 1 layer weight matrix
                 b: {2x1 cell} containing 2 bias vectors

    other:

              name: ''
          userdata: (user information)
```

Gambar 3.6 Perancangan Arsitektur

3.7.1 Algoritma Pelatihan

Pelatihan suatu jaringan dengan algoritma backpropagation meliputi dua tahap : perambatan maju dan perambatan mundur. Selama perambatan maju, tiap unit masukan (x_i) menerima sebuah masukan sinyal ini ke tiap-tiap lapisan tersembunyi z_1,…..,z_p. Tiap unit tersembunyi ini kemudian menghitung aktivasinya dan mengirimkan sinyalnya (z_j) ke tiap unit keluaran. Tiap unit keluaran (y_k) menghitung aktivasinya (y_k) untuk membentuk respon pada jaringan untuk memberikan pola masukan. Selama pelatihan, tiap unit keluaran membandingkan perhitungan aktivasinya y_k dengan nilai targetnya t_k untuk menentukan kesalahan pola tersebut dengan unit itu.

Berdasarkan kesalahan ini, faktor δ_k (k = 1,..,m) dihitung. δ_k digunakan untuk menyebarkan kesalahan pada unit keluaran y_k kembali ke semua unit pada lapisan sebelumnya (unit-unit tersembunyi yang dihubungkan ke y_k). Juga digunakan (nantinya) untuk mengupdate bobotbobot antara keluaran dan lapisan tersembunyi. Dengan cara yang sama, faktor (j = 1,…,p) dihitung untuk tiap unit tersembunyi z_j. Tidak perlu untuk menyebarkan kesalahan kembali ke lapisan masukan, tetapi δ_j digunakan untuk mengupdate bobot-bobot antara lapisan tersembunyi dan lapisan masukan. Setelah seluruh faktor δ ditentukan, bobot untuk semua lapisan diatur secara serentak. Pengaturan bobot w_{jk} (dari unit tersembunyi z_j ke unit keluaran y_k) didasarkan pada faktor δk dan aktivasi z_j dari unit tersembunyi z_j. didasarkan pada faktor δ_j dan dan aktivasi x_i unit masukan. Untuk langkah selengkapnya adalah :

A. Prosedur Pelatihan

Langkah 0 : Inisialisasi bobot. (sebaiknya diatur pada nilai acak yang kecil),

Langkah 1 : Jika kondisi tidak tercapai, lakukan langkah 2-9,

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

Perambatan Maju :

Langkah 3 : Tiap unit masukan (x_i, i = 1,…, n) menerima sinyal x_i dan menghantarkan sinyal ini ke semua unit lapisan di atasnya (unit tersembunyi),

Langkah 4 : Setiap unit tersembunyi (x_i, i = 1,…, p) jumlahkan bobot

sinyal masukannya,

$$z\_in_j = v_{oj} + \sum_{i-1}^{n} x_i v_{ij} \quad (3.8)$$

v_{oj} = bias pada unit tersembunyi j aplikasikan fungsi aktivasinya untuk menghitung sinyal keluarannya, z_j = f (z_in_j), dan kirimkan sinyal ini keseluruh unit pada lapisan diatasnya (unit keluaran).

Langkah 5 : Tiap unit keluaran (y_k, k = 1,…, m) jumlahkan bobot sinyal masukannya,

$$y\_in_k = w_{ok} + \sum_{j-1}^{n} z_j w_{jk} \quad (3.9)$$

w_{ok} = bias pada unit keluaran k dan aplikasikan fungsi aktivasinya untuk menghitung sinyal keluarannya, y_k = f (y_in_k).

Perambatan Mundur :

Langkah 6 : Tiap unit keluaran (y_k, k = 1,…, m) menerima pola target yang saling berhubungan pada masukan pola pelatihan, hitung kesalahan informasinya,

$$\delta_k = (t_k - y_k) f^1(y_{in_k}) \quad (3.10)$$

hitung koreksi bobotnya (digunakan untuk memperbaharui w_{jk} nantinya),

$$\Delta w_{jk} = \alpha \delta_k z_j \quad (3.11)$$

hitung koreksi biasnya (digunakan untuk memperbaharui w_{ok} nantinya), dan kirimkan δ_k ke unit-unit pada lapisan dibawahnya,

Langkah 7 : Setiap unit lapisan tersembunyi (z_j, j = 1,…, p) jumlahkan hasil perubahan masukannya (dari unit-unit lapisan diatasnya),

$$\Delta_{in_j} = \sum_{k-1}^{m} \delta_k w_{jk} \quad (3.12)$$

kalikan dengan turunan fungsi aktivasinya untuk menghitung informasi kesalahannya,

$$\delta_k = \delta\_in_j f^1(z\_in_j) \quad (3.13)$$

hitung koreksi bobotnya (digunakan untuk memperbaharui v_{oj} nanti),

Langkah 8 : Tiap unit keluaran (y_k, k = 1,…, m) update bias dan bobotnya (j = 0,…, p) :

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (3.14)$$

Tiap unit lapisan tersembunyi (z_j, j = 1,…, p) update bias dan bobotnya (I = 0,…,n) :

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad (3.15)$$

Langkah 9 : Test kondisi berhenti.

B. Prosedur Pengujian

Setelah pelatihan, jaringan saraf backpropagation diaplikasikan dengan hanya menggunakan tahap perambatan maju dari algoritma pelatihan. Prosedur aplikasinya adalah sebagai berikut:

Langkah 0 : Inisialisasi bobot (dari algoritma pelatihan).

Langkah 1 : Untuk tiap vektor masukan, lakukan langkah 2-4.

Langkah 2 : for i = 1,…, n : atur aktivasi unit masukan x_i.

Langkah 3 : for j = 1,…, p :

$$z\_in_j = v_{oj} + \sum_{i-1}^{n} x_i v_{ij} \quad (3.16)$$

$$z_j = f(z\_in_j) \quad (3.17)$$

Langkah 4 : for k = 1,…, m :

$$y\_in_k = w_{ok} + \sum_{j-1}^{p} z_j w_{jk} \quad (3.18)$$

$$y_k = f(y\_in_k) \quad (3.19)$$

Langkah 5 : Jika $y_k \geq 0{,}5$ maka y_k = -1, else y_k = 1.

3.7.2 Bobot dan Bias

Pemilihan bobot awal sangat mempengaruhi jaringan syaraf dalam mencapai minimum terhadap nilai error, serta cepat tidaknya proses pelatihan menuju kekonvergenan. Dalam hal ini, pemberian nilai bobot dan bias awal menggunakan bilangan acak kecil yang dilakukan oleh software Matlab. Berikut persamaan bias pada unit tersembunyi aplikasikan fungsi aktivasinya untuk menghitung sinyal keluarannya dan kirimkan sinyal ini keseluruh unit pada lapisan diatasnya (unit keluaran).

$$z\_in_j = v_{oj} + \sum_{i-1}^{n} x_i v_{ij} \quad (3.20)$$

Berikut persamaan bias pada unit keluaran dan fungsi aktivasinya untuk menghitung sinyal keluarannya,

$$y\_in_k = w_{ok} + \sum_{j-1}^{n} z_j w_{jk} \quad (3.21)$$

dengan bobot antara lapisan input dan lapisan tersembunyi akhir (V) dan bias antara lapisan tersembunyi dan lapisan output akhir setelah iterasi seperti terlihat pada Tabel 3.1.

Tabel 3.1. Contoh bobot input akhir dari lapisan input ke lapisan tersembunyi

No	W	Y
1	0.0150	-1
2	0.3557	0
3	0.0778	0
4	0.4943	0
5	0.0598	0
6	0.0711	0
7	0.1325	0
8	0.6622	1

3.7.3 Jumlah Neuron pada Lapisan Tersembunyi

Perancangan arsitektur backpropagation pertama adalah menentukan jumlah hidden layer dan menentukan banyaknya neuron dalam setiap hidden layer. Arsitektur jaringan yang digunakan dalam skripsi ini adalah arsitektur jaringan dengan satu hidden layer. Menurut teori, arsitektur ini disebut arsitektur jaringan layar jamak. Banyaknya neuron hidden layer ditentukan dengan cara trial and error, dalam arti hasil pembelajaran yang tercepat dan terbaik itulah yang akan menentukan banyaknya neuron hidden layer tersebut. Mengenai jumlah banyaknya neuron hidden layer yang dibutuhkan, tidak ada ketentuan khusus karena tidak ada teori yang dengan pasti dapat dipakai (Mirawanti et all., 2010).

3.7.4 Error Goal (Kinerja Tujuan)

Kinerja tujuan adalah target nilai fungsi kinerja. Iterasi akan dihentikan apabila nilai fungsi kinerja kurang dari atau sama dengan kinerja tujuan (Kusumadewi, 2004:134).

Error goal atau galat ditentukan untuk membandingkan dengan galat pada jaringan saat pelatihan. Jaringan akan konvergen ketika error jaringan lebih kecil dari error goal. Dalam skripsi ini ditentukan error goal atau toleransi sebesar 0.

Segmen program 3.1 Error Goal

1. `net.trainParam.goal = 0;`

keterangan :

`net` = jaringan backpropagation yang terdiri dari n layer.

`trainParam.goal` = fungsi perhitungan pada error MSE (Mean Square Error).

`0` = nilai error yang digunakan untuk proses train dan uji.

Pada segmen program 3.1 menggunakan "`net.trainParam.goal = 0;`" untuk dapat melatih jaringan meminimalkan error. Pada fungsi perhitungan nilai error ini menggunakan nilai error goal atau toleransi sebesar 0.

3.7.5 Learning Rate (Laju Pembelajaran)

Semakin besar nilai learning rate (α) akan berimplikasi pada semakin besarnya langkah pembelajaran. Jika learning rate diset terlalu besar, maka algoritma akan menjadi tidak stabil. Sebaliknya, jika learning rate diset terlalu kecil, maka algoritma akan konvergen dalam jangka waktu yang sangat lama (Kusumadewi, 2004: 134).

Nilai α terletak antara 0 dan 1 ($0 \leq \alpha \leq 1$). Jika harga α semakin besar, maka iterasi yang dipakai semakin sedikit. Hal ini menyebabkan pola yang sudah benar menjadi rusak sehingga pemahaman menjadi lambat. Nilai learning rate tidak dapat ditentukan secara pasti sehingga perlu dilakukan trial and error untuk mendapatkan nilai learning rate yang dapat menghasilkan iterasi tercepat dalam mencapai konvergen (Amin, 2012).

3.7.6 Fungsi Aktivasi

Dalam jaringan syaraf tiruan, fungsi aktivasi digunakan untuk menentukan keluaran suatu neuron. Fungsi aktivasi yang digunakan dalam penerapan JST untuk deteksi kematangan buah pepaya, yaitu fungsi aktivasi tansig. Fungsi ini dipilih karena pada unit output dirancang menampilkan beberapa keputusan, yaitu apakah pepaya mentah dengan nilai -1, pepaya setengah matang dengan nilai 0 dan pepaya matang dengan nilai 1.

Segemen program 3.2 Fungsi Aktivasi

```
1.  net = newff(minmax(a'),[4 1],{'tansig''tansig'},'traingdm');
```

Keterangan :

`net` = jaringan backpropagation yang terdiri dari n layer.

`Newff` = perintah untuk membuat jaringan.

`(minmax(a')` = matrik ordo yang berisi nilai minimum dan maksimum elemen masukannya.

`[4]` = jumlah neuron tersembunyi dengan fungsi aktifasi '*tansig*'

`[1]` = jumlah neuron output dengan fungsi aktifasi '*Traingdm'*

`'tansig'` = ini adalah fungsi default sigmoid bipolar.

`'traingdm'` = ini adalah metode backpropagation yang dipercepat dengan momentum.

Pada segmen program 3.2 menggunakan "`net = newff(minmax(a')`" untuk dapat membangun sebuah jaringan syaraf tiruan dengan nilai minimum dan maxsimum sesuai dengan pola 'a'. 'a' adalah data pola nilai yang digunakan pada saat pelatihan. Kemudian "`[4 1] {'tansig','tansig',}'traingdm'`" adalah jumlah neuron yang tersembunyi pada jaringan yaitu sebanyak 4 neuron. Kemudian menggunakan fungsi tansig yaitu fungsi default sigmoid bipolar. Kemudian angka 1] adalah jumlah neuron output dengan fungsi aktivasi trainlm, trainlm adalah fungsi aktivasi backpropagation yang dipercepat dengan momentum.

fungsi ini akan membawa nilai input pada ooutput dengan menggunakan rumus *hyperbolic tangen sigmoid*. Nilai maksimal output dari fungsi ini adalah 1 dan minimal -1.

Algoritma dan fungsi ini adalah

$$a = tansig(n) = \frac{2}{1+\exp(-2*n)} - 1 \quad (3.22)$$

3.7.7 Maksimum Epoch

Jumlah epoh maksimum yang boleh dilakukan selama proses pelatihan. Iterasi akan dihentikan apabila epoh melebihi maksimum epoh.

Segmen program 3.3 Maksimum Epoch

```
1.  net.trainParam.epochs = 1500;
2.  net=train(net,a',t);
3.  save net.mat net
```

keterangan :

`net` = jaringan backpropagation yang terdiri dari n layer.

`trainParam.epochs` = fungsi perhitungan pada Iterasi.

`1500` = jumlah Iterasi maksimum.

`net=train` = fungsi pelatihan backpropagation.

`(net,a',t);` = pola masukan a' terhadap target pada jaringan net.

`save net.mat net` = perintah kode program untuk menyimpan hasil training.

Pada segmen program 3.3 menggunakan "`trainParam.epochs`" untuk fungsi perhitungan pada saat iterasi. Pada penelitian ini maksimum proses iterasi menggunakan 1500 iterasi. Kemudian "`net=train`" adalah fungsi untuk memanggil pelatihan backpropagation untuk melatih jaringan, kemudian "`(net,a',t)`" adalah nilai pola masukan terhadap target pada jaringan net. Kemudian "`save net.mat net`" adalah fungsi untuk menyimpan hasil training ke bentuk file.mat pada matlab.

3.7.8 Deteksi Buah Pepaya

Pada proses ini maka citra yang dihasilkan adalah citra pepaya yang terdeteksi berdasarkan tingkat kematangan tertentu. Dari hasil deteksi nilai masing-masing RGB dan akurasi yang didapat akan diketahui sesuai tingkat kematangan buah pepaya.

Segmen program 3.4 Output Deteksi

```
1.  y=sim(net,c')
2.  y=round(y)
```

keterangan :

`sim` = fungsi yang digunakan untuk menghitung keluaran jaringan

untuk dicocokan pada target.

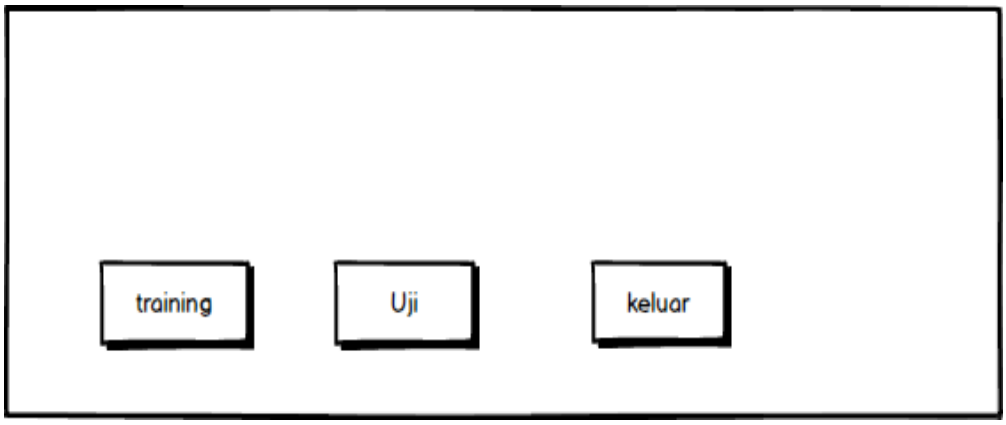
`y=round(y)` = Output yang akan ditampilkan.

Pada segmen program 3.4 menggunakan "`y=sim(net,c')`" yaitu fungsi ini diginakan untuk menghitung nilai output jaringan backpropagation untuk dicocokkan pada target. Target mentah bernilai (-1), target setengah matang bernilai (0), dan target matang bernilai (1). Kemudian "`y=round(y)`" adalah output yang akanditampilkan pada command windows setelah proses identifikasi.

3.8 Perencanaan interface aplikasi model backpropagation

Perancangan sistem untuk model backpropagation secara garis besar proses dapat dikelompokkan menjadi tiga bagian yaitu menu, proses pelatihan data (pembelajaran backpropagation) dan identifikasi kematangan buah pepaya. Proses pelatihan berguna untuk melatih sistem dengan memasukkan data-data inputan ke dalam sistem Neural Network kemudian data tersebut diolah dengan menggunakan metode backpropagation.

3.8.1 Interface Menu



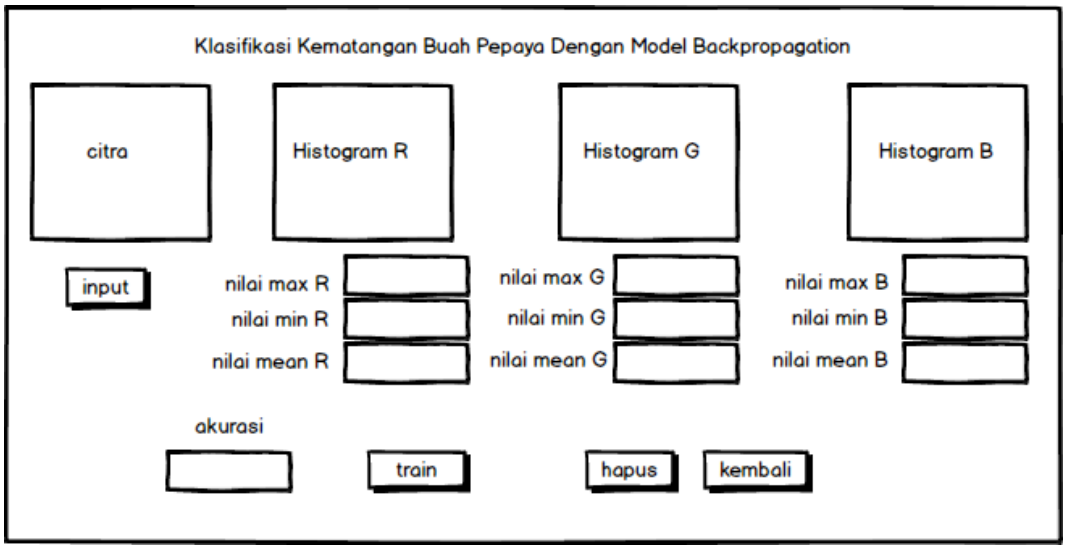
Gambar 3.7. rancangan interface menu aplikasi model backpropagation

Rancangan layar Menu aplikasi sistem GUI ini tidak dapat digunakan secara langsung. Rancangan sistem backpropagation dengan GUI dijelaskan sebagai berikut:

1. Tombol training merupakan tombol yang digunakan untuk masuk ke dalam aplikasi data training.
2. Tombol uji merupakan tombol yang digunakan untuk masuk ke dalam aplikasi data uji.

3. Tombol keluar merupakan tombol yang digunakan untuk keluar dari aplikasi backpropagation.

3.8.2 Interface Data Training



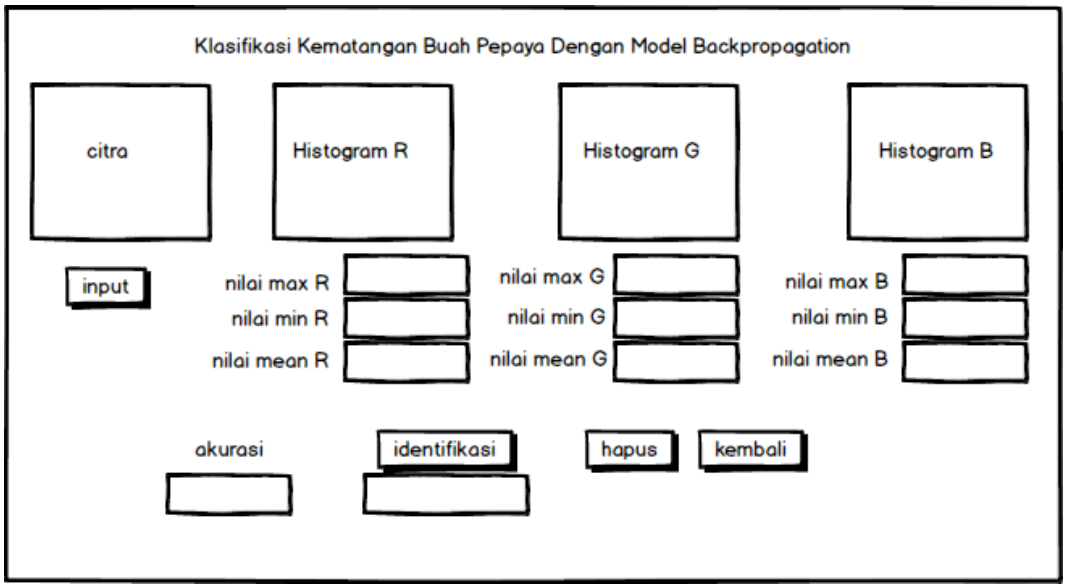
Gambar 3.8 rancangan interface data training aplikasi model backpropagation

Rancangan layar aplikasi sistem GUI ini tidak dapat digunakan secara langsung. Karena ini hanya untuk perencanaan sistem yang akan dibangun pada aplikasi klasifikasi kematangan buah pepaya dengan model backpropagation. rancangan sistem backpropagation dengan GUI dijelaskan sebagai berikut:

1. Bagian satu merupakan judul dari rancangan sistem backpropagation yang dibangun dengan GUI.
2. Tombol input merupakan rancangan tombol untuk mengmbil gambar buah pepaya pada komputer yang sudah terdata.
3. Gambar buah pepaya yang telah diambil dengan menggunakan tombol input yang akan ditampilkan pada layar citra yaitu gambar buah pepaya hasil input.
4. Hasil input gambar histogram R akan ditampilkan di layar histogram R.
5. Hasil input gambar histogram G akan ditampilkan di layar histogram G.
6. Hasil input gambar histogram B akan ditampilkan di layar histogram B.
7. Hasil input nilai max, min dan mean untuk R gambar akan ditampilkan pada text box max, min, mean R.
8. Hasil input nilai max, min dan mean untuk G gambar akan ditampilkan pada text box max, min, mean G.

9. Hasil input nilai max, min dan mean untuk B gambar akan ditampilkan pada text box max, min, mean B.
10. Label akurasi untuk menampilkan hasil akurasi dari inputan nilai masing-masing RGB setelah di latih.
11. Tombol train untuk melatih data yang akan di uji.
12. Tombol hapus merupakan rancangan tombol untuk menghapus seluruh data yang sudah di training.
13. Tombol kembali merupakan rancangan tombol untuk kembali ke interface menu awal.

3.8.3 Interface Data Uji



Gambar 3.9 rancangan interface data uji aplikasi model backpropagation

Rancangan layar aplikasi sistem GUI ini tidak dapat digunakan secara langsung. Rancangan sistem backpropagation dengan GUI dijelaskan sebagai berikut:

1. Bagian satu merupakan judul dari rancangan sistem backpropagation yang dibangun dengan GUI.
2. Tombol input merupakan rancangan tombol untuk mengmbil gambar buah pepaya yang ada pada komputer yang sudah terdata.
3. Gambar buah pepaya yang telah diambil dengan menggunakan tombol input yang akan ditampilkan pada layar citra yaitu gambar buah pepaya hasil input.
4. Hasil input gambar histogram R akan ditampilkan di layar histogram R.

5. Hasil input gambar histogram G akan ditampilkan di layar histogram G.
6. Hasil input gambar histogram B akan ditampilkan di layar histogram B.
7. Hasil input nilai max, min dan mean untuk R gambar akan ditampilkan pada text box max, min, mean R.
8. Hasil input nilai max, min dan mean untuk G gambar akan ditampilkan pada text box max, min, mean G.
9. Hasil input nilai max, min dan mean untuk B gambar akan ditampilkan pada text box max, min, mean B.
10. Label akurasi dari inputan nilai masing-masing RGB setelah di uji.
11. Text box akurasi untuk menampilkan hasil akurasi dari inputan nilai masing-masing RGB setelah di uji.
12. Tombol identifikasi untuk menguji data untuk menentukan kematangan buah pepaya.
13. Text box identifikasi untuk menampilkan hasil identifikasi hasil uji dari inputan nilai masing-masing RGB setelah di uji.
14. Tombol hapus merupakan rancangan tombol untuk menghapus seluruh data yang sudah di uji.
15. Tombol kembali merupakan rancangan tombol untuk kembali ke interface menu awal.