

**SORTASI BUAH KOPI UNTUK MENENTUKAN MUTU BUAH  
KOPI MENGGUNAKAN JARINGAN SYARAF TIRUAN  
METODE *BACKPROPAGATION* DENGAN MATLAB**

(Skripsi)

Oleh

**Yeni Apriyana**



**JURUSAN TEKNIK ELEKTRO  
FAKULTAS TEKNIK  
UNIVERSITAS LAMPUNG  
BANDAR LAMPUNG  
2018**

# **BAB I**

## **PENDAHULUAN**

### **1.1. Latar Belakang**

Kopi merupakan salah satu komoditas perkebunan yang banyak diperdagangkan dan masuk dalam komoditas strategis di Indonesia. Provinsi Lampung merupakan salah satu daerah yang menghasilkan produksi kopi yang cukup tinggi. Berdasarkan data dari Badan Pusat Statistik Perkebunan Indonesia (BPSPI) tahun 2015 – 2017, luas area dan produksi kopi tahun 2015 Lampung memiliki luas area perkebunan kopi sebesar 161.693 Ha dengan produksi mencapai 110.318 Ton, tahun 2016 dengan luas area 161.416 Ha dengan hasil produksi 110.354 Ton, dan pada tahun 2017 dengan luas area 162.020 Ha dengan produksi mencapai 110.368 Ton [1]. Berdasarkan data tersebut menunjukkan bahwa Provinsi Lampung memiliki potensi untuk melakukan pengembangan agroindustri kopi dengan semakin meningkatnya produksi kopi yang dihasilkan. Pengolahan kopi harus dilakukan dengan tata cara yang tepat agar menghasilkan kopi dengan kualitas dan mutu yang baik.

Proses pengolahan buah kopi menjadi biji kopi yang dilakukan oleh industri maupun petani dibagi menjadi dua tahapan yaitu pengolahan dengan proses basah dan pengolahan dengan proses kering. Pengolahan kopi dengan proses basah dimulai dari panen, sortasi buah kopi, pengupasan kulit buah merah, fermentasi, pencucian, pengeringan, pengupasan kulit tanduk dan kulit ari, sortasi biji kopi

pengemasan, dan penyimpanan. Sedangkan pengolahan kopi dengan proses kering terdiri dari panen, sortasi buah kopi, pengeringan, pengupasan, sortasi biji kopi, pengemasan, dan penyimpanan. Produk yang dihasilkan dari kopi tidak hanya sebagai kopi bubuk biasa tetapi juga dapat diturunkan kembali menjadi kopi instan bebas kafein, kopi dengan percampuran teh, es kopi instant, kopi putih, kopi dengan rasa buah – buahan dan masih banyak lagi.

Semakin meningkatnya permintaan pasar mengenai kopi menyebabkan banyak terjadi keterhambatan ketersediaan kopi yang diminta karena petani masih menggunakan peralatan manual serta dengan cara yang tradisional dalam proses sortasi buah kopi. Sedangkan pada tahap proses pengolahan kopi sortasi buah kopi sangat penting pada saat buah kopi dipanen. Buah kopi yang telah disortasi tidak dapat disimpan terlalu lama karena apabila penundaan pengolahan bisa memicu reaksi kimia yang akan menurunkan mutu kopi itu sendiri. Pemisahan buah dilakukan untuk membedakan kualitas biji kopi yang akan dihasilkan nanti untuk memilih buah yang superior (masak dan seragam) dari buah inferior (cacat, hitam, pecah, berlubang dan terserang hama/penyakit)

Tingkat kematangan buah kopi dapat dilihat dari warna kulitnya, apabila buah kopi telah berwarna merah berarti menandakan bahwa buah kopi tersebut telah matang penuh. Namun karena berbagai alasan, sortasi buah kopi secara manual sangat dipengaruhi oleh subjektivitas petani kopi sehingga menyebabkan tidak spesifiknya proses klasifikasi buah kopi. Hal tersebut menyebabkan rendahnya mutu kopi bubuk hasil olahan petani salah satunya disebabkan oleh proses sortasi buah kopi yang masih menggunakan cara manual seperti melakukan sortasi secara langsung oleh manusia, sehingga menyebabkan semakin banyaknya waktu dan sumber daya manusia yang dibutuhkan dalam proses sortasi buah kopi tersebut.

Pada dasarnya terdapat tingkat kematangan buah kopi yaitu apabila buah kopi berwarna hijau dan hijau kekuningan menandakan bahwa kondisi buah kopi masih muda dan tidak disarankan untuk dipetik, warna kuning kemerahan menunjukkan buah kopi sudah mulai matang namun belum memiliki aroma dan postur yang baik, warna merah penuh, menunjukkan buah telah matang sempurna atau memiliki aroma yang baik, dan yang terakhir warna merah tua atau kehitam hitaman menandakan buah kopi tersebut sudah terlalu matang serta aroma dan posturnya yang mulai menurun [2].

Melihat pentingnya proses sortasi pada buah kopi untuk menghasilkan kopi yang bermutu baik yang berasal pada buah kopi yang sehat dan berwarna merah maka perlu dibangun sebuah sistem. Dengan perkembangan teknologi informasi saat ini memungkinkan untuk melakukan identifikasi buah berdasarkan ciri warna dengan bantuan komputer [3]. Pada cara komputasi tersebut dilakukan dengan pengamatan visual tidak langsung, dengan menggunakan kamera sebagai pengolahan citra dari gambar yang di foto (*image processing*) yang akan diolah menggunakan perangkat lunak komputer [4].

Berdasarkan uraian diatas maka dibangun suatu sistem identifikasi kematangan buah kopi dengan pengolahan citra serta pada proses indentifikasi buah kopi menggunakan Jaringan Syaraf Tiruan (JST) dengan metode pembelajaran *backpropagation*. Penggunaan pengolahan citra sendiri digunakan untuk melakukan proses *pre-processing* sebelum memasuki tahap identifikasi gambar. JST *backpropagation* merupakan topologi yang cukup populer dan banyak dipakai untuk berbagai aplikasi terutama dalam pengenalan pola[5].

JST *backpropagation* merupakan pelatihan jaringan untuk mengetahui kemampuan jaringan dalam mengenali pola yang digunakan selama pelatihan serta kemampuan jaringan dalam melakukan respon yang benar

terhadap pola masukan yang serupa dengan pola yang akan dipakai selama proses pelatihan [6]. Sistem ini dibangun dengan menggunakan inputan gambar buah kopi kemudian dilakukan pengolahan citra untuk menerangkan kontras warna gambar kemudian dilakukan pembuangan background yang kemudian gambar akan di ambil nilai R, G dan B nya sebagai inputan dalam proses klasifikasi *backpropagation* yang nantinya akan mendapatkan 3 outputan dari masing-masing warna berupa nilai biner 1/0.

## **1.2. Tujuan Penelitian**

Tujuan dari penelitian ini sebagai berikut :

1. Menguji tingkat keberhasilan kualitas sortasi menggunakan JST *backpropagation*.
2. Sebagai studi penelitian mengenai jaringan syaraf tiruan dengan metode *backpropagation*.
3. Mewujudkan visi teknik informatika unila dalam kekhasan pada bidang *ICT in Agriculture*.

## **1.3. Rumusan Masalah**

Rumusan masalah yang mendasari penelitian ini sebagai berikut :

1. Bagaimana membangun sebuah sistem pengolahan citra digital dalam menentukan kualitas buah kopi pada warna menggunakan metode JST *backpropagation* menggunakan MATLAB?
2. Apakah sistem yang dibangun sudah dapat melakukan sortasi buah kopi sesuai dengan kriteria yang di harapkan?

#### **1.4. Batasan Masalah**

Batasan masalah pada penelitian ini sebagai berikut :

1. Menggunakan aplikasi matlab sebagai sistem pengolahan citra dan JST.
2. Menggunakan gambar buah kopi sebagai sample dengan ukuran 500 x 500 pixel dengan berbagai warna yang tersebar pada setiap sample.
3. Tidak membahas pemrograman matlab secara mendalam.

#### **1.5. Manfaat Penelitian**

Manfaat dari penelitian ini sebagai berikut :

1. Membantu petani kopi dengan membangun sistem sortasi buah kopi.
2. Sistem dapat mengefisiensikan waktu dalam proses sortasi buah kopi.
3. Penulis dapat lebih memahami mengenai pengolahan citra digital dan JST terutama pada metode *backpropagation*.

#### **1.6. Sistematika Penulisan**

Pada penulisan laporan ini untuk mempermudah penulisan dan pemahaman mengenai materi maka tulisan ini dibagi menjadi lima bab, yaitu :

##### **BAB I      Pendahuluan**

Pada bab ini diuraikan hal-hal yang berhubungan dengan latar belakang, tujuan penelitian, rumusan masalah, batasan masalah, metode penelitian, dan sistematika penulisan.

##### **BAB II     Tinjauan pustaka**

Bab ini merupakan landasan teori dari penyusunan penulisan yaitu prinsip, pengetahuan, rumus, dan teori penunjang

tentang kopi, pengolahan citra, JST *backpropagation*, MATLAB dan lain-lain.

### **BAB III Metode penelitian**

Bab ini berisi mengenai waktu penelitian, alat dan bahan penelitian , tahapan penelitian secara terperinci.

### **BAB IV Hasil dan PEMBAHASAN**

Pada bagian ini memaparkan mengenai perancangan sistem menggunakan *backpropagation*, data hasil penelitian dan pembahasan dari data-data yang didapatkan saat pengujian.

### **BAB V Simpulan dan Saran**

Berisikan kesimpulan dari hasil penelitian dan saran-saran mengenai perbaikan dan pengembangan penelitian ini .

**Daftar pustaka**

**Lampiran**

## **BAB II**

### **TINJAUAN PUSTAKA**

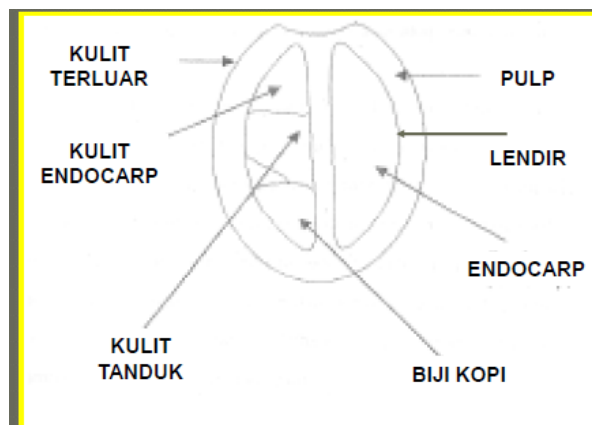
#### **2.1. Kopi**

Kopi merupakan tanaman yang termasuk dalam famili *Rubiaceae*. Pada dasarnya kopi memiliki beberapa golongan, namun yang paling sering dibudidayakan yaitu *Coffea Arabica*, *Coffea Robusta*, dan *Coffea Liberica*. Tanaman kopi berasal dari Abessinia yang tumbuh di dataran tinggi dengan nama lain kopi yaitu *Coffea sp.* Berikut ini merupakan klasifikasi menurut Cronquist (1981) sebagai berikut [7] :

Regnum	: Plantae
Divisio	: Magnoliophyta
Sub Divisio	: Spermatophyta
Class	: Magnoliopsida
Sub Class	: Asteridae
Order	: Rubiales
Family	: Rubiaceae
Genus	: Coffea

Tanaman kopi pada umumnya berbunga setelah berumur sekitar dua tahun. Pada buah kopi terdiri dari daging buah dan biji. Pada bagian daging biji buah kopi terdiri dari 3 bagian yaitu lapisan kulit luar (*exocarp*), lapisan daging buah (*mesocarp*), dan lapisan kulit tanduk (*endoscarp*). Berikut ini merupakan gambaran struktur anatomi buah kopi yaitu :





Gambar 2.1. Struktur anatomi buah kopi [7].

Pada umumnya buah kopi yang masih muda berwarna hijau, tetapi saat mulai tua menjadi kuning dan apabila telah masak warnanya menjadi merah. Kulit buah kopi sangatlah tipis dan mengandung klorofil serta zat-zat warna lainnya. Pada daging buah terdiri dari 2 bagian yaitu bagian luar yang lebih tebal dan keras dan pada bagian dalamnya yang bersifat seperti gel atau lendir. Pada lapisan lendir ini terdapat 85% air dalam bentuk terikat, dan 15%nya merupakan bahan koloid yang tidak mengandung air. Pada bagian ini bersifat koloid hidrofilik yang terdiri dari  $\pm 80\%$  pectin dan  $\pm 20\%$  gula. Pada biji kopi kering mempunyai komposisi yaitu 12% air, 13% protein, 12% lemak, 9% gula, 1-1,5% *caffeine* (arabika), 2-2,5% (robusta), 9% *caffetanic acid cellulose* dan sejenisnya 35%, 4% abu, zat-zat lainnya yang larut dalam air 5% [8].

Pada umumnya buah kopi memiliki dua butir biji, namun pada kenyataannya terdapat pula buah kopi yang tidak menghasilkan biji atau hanya menghasilkan satu butir biji. Pada buah kopi biji kopi memiliki bagian-bagian penting yang terdiri dari kulit biji dan lembaga. Secara morfologi biji kopi berbentuk bulat atau oval seperti telur, memiliki warna putih kotor, dan bertekstur keras [9].

## 2.2. Proses Pengolahan Biji Kopi

Menurut Rahajo (2012) , kopi yang telah dipetik sebaiknya segera diolah lebih lanjut dan tidak dianjurkan untuk dibiarkan begitu saja selama kurun waktu lebih dari 12-20 jam. Apabila buah kopi tidak segera diolah dalam jangka waktu tertentu maka kopi akan mengalami fermentasi dan proses kimia lainnya yang dapat mengakibatkan penurunan mutu kopi tersebut. Namun bila kopi benar-benar belum dapat diolah, maka kopi harus direndam terlebih dahulu dalam air bersih yang mengalir [10]. Proses pengolahan kopi pada dasarnya terbagi menjadi dua yaitu proses olahan kering (*dry process*) dan proses olahan basah (*wet process*) [11].

### 2.2.1. Pengolahan Cara Kering

Pada metode pengolahan kopi dengan cara kering banyak digunakan untuk pengolahan ditingkat petani dengan lahan yang tidak luas atau dengan kapasitas olahan yang kecil. Pada perkebunan yang memiliki lahan lebih besar pengolahan kopi dengan cara kering hanya khusus untuk kopi dengan buah yang berwarna hijau, kopi yang mengambang dan kopi yang terserang penyakit. Berikut ini merupakan tahapan pengolahan kopi dengan cara kering [11] :



Gambar 2.2 Alur pengolahan kopi secara kering (*dry process*).

### 2.2.2. Pengolahan Cara Basah

Pengolahan kopi dengan cara basah yaitu dengan cara buah kopi yang telah dipetik selanjutnya dimasukkan kedalam *pulper* untuk melepaskan kulitnya kemudian dibiarkan ke bak dan direndam selama beberapa hari untuk fermentasi. Kemudian setelah direndam buah kopi tersebut di cuci bersih dan dikeringkan. Pengeringan buah kopi dilakukan dengan dijemur dibawah sinar matahari atau menggunakan mesin pengering. Berikut ini merupakan tahapan pengolahan buah kopi dengan cara basah [11] :



Gambar 2.3. Proses pengolahan buah kopi secara basah  
(*wet process*).

### 2.3. Matlab

Matlab merupakan salah satu alat yang digunakan untuk komputasi matrik. Matlab adalah salah satu perangkat lunak yang melibatkan penggunaan matriks dan vektor. Fungsi-fungsi pada toolbox dibangun untuk

memudahkan pengguna dalam perhitungan tersebut [6]. Pada fungsi matlab terdapat model Jaringan Syaraf Tiruan (JST) yang menggunakan manipulasi matriks atau vektor dalam iterasinya. Maka matlab dianggap perangkat lunak yang cocok digunakan sebagai perancangan sistem ini karena menggunakan metode JST *Backpropagation*.

## 2.4. Penskalaan

Penskalaan atau *Scaling* merupakan sebuah operasi geometri memberika efek memperbesar (*zoomin*) atau memperkecil (*zoomout*) sesuai dengan variabel penskalaan citranya. Proses penskalaan dilakukan dengan rumus :

$$P_o = S_p \times P_i \dots\dots\dots [12]$$

$$L_o = S_l \times L_i \dots\dots\dots [12]$$

Pada rumus tersebut ( $P_i, L_i$ ) merupakan ukuran citra input, sedangkan ( $P_o, L_o$ ) merupakan ukuran citra output, dan ( $S_p, S_l$ ) merupakan variabel *scaling* yang diinginkan. Apabila variabel *scaling* bernilai lebih besar dari 1 maka hasil *scaling* akan memperbesar ukuran citra, namun apabila variabel *scaling* lebih kecil dari 1 maka hasilnya akan memperkecil ukuran citra [12].

## 2.5. Paint

Microsoft Paint merupakan editor grafis sederhana yang disediakan oleh microsoft yang dimiliki semua versi pada Microsoft Windows [13]. Paint merupakan program yang berfungsi untuk membuka, mengedit dan menyimpan file dalam berbagaimacam format file seperti .bitma, JPEG, GIF dan lainnya. Pada penelitian ini paint digunakan sebagai resize gambar dengan ukuran sama yaitu 500x500 pixel.

## 2.6. Pengolahan Citra Digital

Pengolahan citra digital merupakan suatu pemrosesan citra atau *image processing* dengan kegiatan memperbaiki kualitas citra agar mudah dikenali oleh manusia atau komputer sesuai dengan tujuan yang diharapkan [14]. Menurut RD. Kusumanto dan Alan Novi Tompunu (2011) , pengolahan citra digital (*Digital Image Processing*) merupakan disiplin ilmu yang mempelajari mengenai teknik-teknik mengolah citra (foto) maupun gambar bergerak (berasal dari *webcam*) dengan pengolahan citra/gambar dilakukan secara digital menggunakan komputer [15]. Pada jurnal penelitian Dila Deswari, Hendrick dan Derisma (2010), pengolahan citra digital merupakan ilmu yang mempelajari hal-hal yang berkaitan dengan perbaikan kualitas citra, dengan kata lain pengolahan citra merupakan kegiatan memperbaiki kualitas citra agar dapat dengan mudah diterjemahkan oleh manusia atau mesin (komputer) [16].

Citra digital umumnya dua dimensi (2D) yang dinyatakan dengan matriks dengan jumlah elemen yang berhingga. Fungsi dua dimensi  $f(x,y)$  di mana  $x$  dan  $y$  adalah koordinat spasial (*plane*) dan  $f$  adalah nilai intensitas warna pada koordinat  $x$  dan  $y$ . Nilai  $f$ ,  $x$  dan  $y$  semuanya merupakan nilai berhingga. Citra digital merupakan representasi piksel-piksel dalam ruang 2D yang dinyatakan dalam matriks berukuran  $N$  baris dan  $M$  kolom, dengan elemen matriks citra disebut piksel (*picture elemen, image elemen*) atau elemen terkecil dari sebuah citra [14].

### 2.6.1. Representasi Warna

Pada dasarnya mata manusia memiliki persepsi warna yang merupakan proses subjektif di mana syaraf visual otak merespon ketika cahaya masuk. Persepsi warna pada pengolahan citra tergantung pada tiga faktor, yaitu *spectral reflectance* (menentukan

bagaimana suatu permukaan memantulkan warna), *spectral content* (kandungan warna dari cahaya yang menyinari permukaan) dan *spectral response* (kemampuan merespon warna dari sensor dalam *imaging system*) [17].

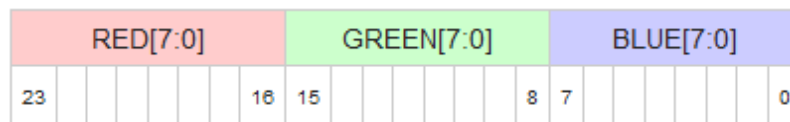
Represensi warna terdiri dari tiga unsur warna utama yaitu merah (*red*), hijau (*green*), dan biru (*blue*) atau disingkat dengan warna RGB. Gabungan dari ketiga warna tersebut membentuk warna-warna lainnya berdasarkan intensitas dari setiap warna tersebut dengan intensitas maksimal, dan warna hitam merupakan gabungan dari ketiga warna tersebut dengan intensitas minimal. Representasi warna dikembangkan dengan tujuan untuk memodelkan, menghitung dan memvisualisasikan informasi warna sehingga dapat dengan mudah komputer atau sistem digital lainnya untuk memproses informasi warna dan membedakan warna seperti halnya sistem visual manusia.

#### **2.6.2. Format File Citra JPEG (.jpg)**

Pada pengolahan citra terdapat format file citra standar yang digunakan yang terdiri dari beberapa jenis. Format file citra digunakan untuk menyimpan citra dalam sebuah file. Berikut ini beberapa format yang digunakan pada penelitian ini. JPEG atau .jpg merupakan salah satu format file citra yang banyak digunakan saat ini terutama untuk melakukan transmisi citra. Format .jpg digunakan sebagai tempat penyimpanan citra hasil kompresi dengan metode JPEG [12].

### 2.6.3. Ruang Warna RGB

Ruang warna RGB adalah ruang warna standar yang didasarkan pada hasil akuisisi frekuensi warna oleh sensor elektronik. Bentuk keluaran dari sensor ini adalah berupa sinyal analog, yang kemudian intensitas amplitudonya di digitalisasi dan dikodekan dalam 8 bit untuk setiap warnanya.. Setiap warna yang tampak merupakan kombinasi dari tiga komponen red, green dan blue. Gabungan tiga warna ini akan menggabungkan warna lain, selain itu R,G, dan B memiliki intensitas yang sama ,setiap titik atau pixel pada citra berwarna memiliki tiga komponen warna red, green dan blue yang masing-masing umumnya dikodekan dengan 8 bit, atau total ketiganya adalah  $3 \times 8 = 24$  bit . [14].



Gambar 2.4. R,G, dan B kode 24 bit [16].

Pada perhitungan nilai R,G, dan B sendiri secara matematis dapat dihitung dengan rumus R :

$$R, G, \text{ dan } B = (R * 65536) + (G * 256) + B \dots\dots\dots [17]$$

Keterangan :

R = Pembulatan kebawah (RGB / 65536/ 256\*256)

G = Pembulatan kebawah (RGB / 256) Modulus

B = RGB Modulus 256.

Modulus merupakan sisa pembagian dari suatu bilangan.

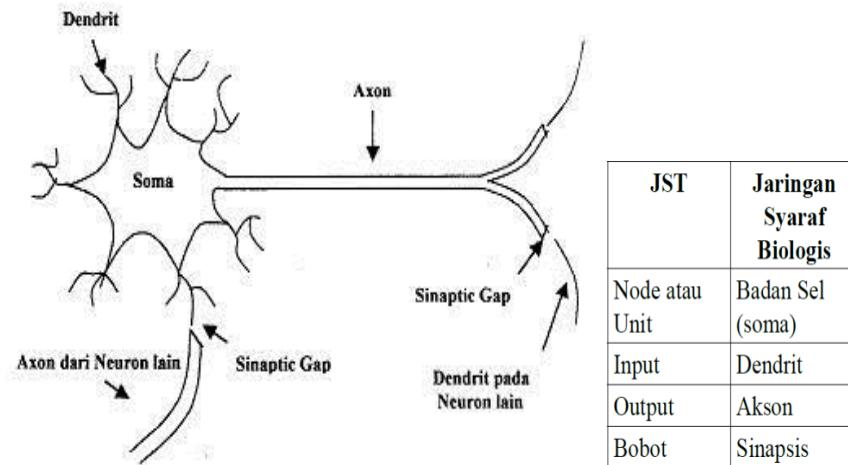
#### **2.6.4. Pemerataan Kontras**

Penyamaan histogram adaptif merupakan pemerataan histogram adaptif dengan kontras terbatas atau *Contrast-Limited Adaptive Histogram Equalization* (CLAHE) dengan menggunakan fungsi *adapthisteq*. Histogram sendiri merupakan grafik yang mengindikasikan jumlah kemunculan setiap level keabuan pada suatu citra [18]. *Adapthisteq* beroperasi pada daerah kecil pada gambar yang disebut ubin, *adapthisteq* berguna sebagai peningkatan kontras setiap ubin sehingga histogram dari wilayah output menjadi lebih terang dan jelas [19].

### **2.7. Jaringan Syaraf Tiruan**

Jaringan syaraf tiruan (JST) merupakan sistem pemrosesan informasi yang memiliki karakteristik mirip dengan jaringan syaraf biologi. JST dibentuk sebagai generalisasi model matematika dari jaringan syaraf biologi sehingga menjadikan JST cocok untuk menyelesaikan masalah dengan tipe yang menyerupai otak manusia. Otak manusia sendiri terdiri dari neuron-neuron dan penghubung yang disebut sinapsis. Neuron bekerja berdasarkan implus/ sinyal yang diberikan pada neuron. Neuron meneruskannya pada neuron lain. Neuron memiliki 3 komponen penting yaitu dendrit, soma dan axon. Dendrit menerima sinyal dari neuron lain, sinyal tersebut berupa impuls elektrik yang dikirim melalui celah sinaptik melalui proses kimiawi. Sinyal tersebut dimodifikasi (diperkuat/diperlemah) di celah sinaptik. Soma menjumlahkan semua sinyal-sinyal yang masuk. Jika jumlah tersebut cukup kuat dan melebihi batas ambang (threshold), maka sinyal tersebut akan diteruskan ke sel lain melalui axon. Frekuensi penerusan sinyal berbeda-beda antara satu sel dengan yang lain [6].

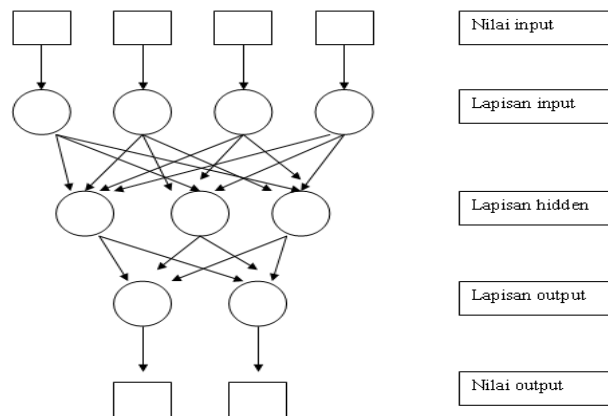




Gambar 2.5. Jaringan Syaraf Biologi dan JST

### 2.7.1. Arsitektur Jaringan

Beberapa arsitektur jaringan yang sering dipakai dalam jaringan syaraf tiruan yaitu jaringan layar tunggal, jaringan layar jamak dan jaringan kompetitive. Pada JST *backpropagation* menggunakan arsitektur jaringan jamak (*multi layer*). Jaringan *multi layer* merupakan perluasan dari layar tunggal. Dalam jaringan ini, selain unit input dan output, ada unit-unit lain (sering disebut layar tersembunyi/ *hidden layer*). Seperti pada unit input dan output, unit-unit dalam satu layar tidak saling berhubungan, memiliki satu atau lebih lapisan input, satu atau lebih lapisan output dan lapisan tersembunyi. Dapat menyelesaikan masalah yang lebih kompleks karena lebih akurat dan fungsi pembelajaran lebih rumit.



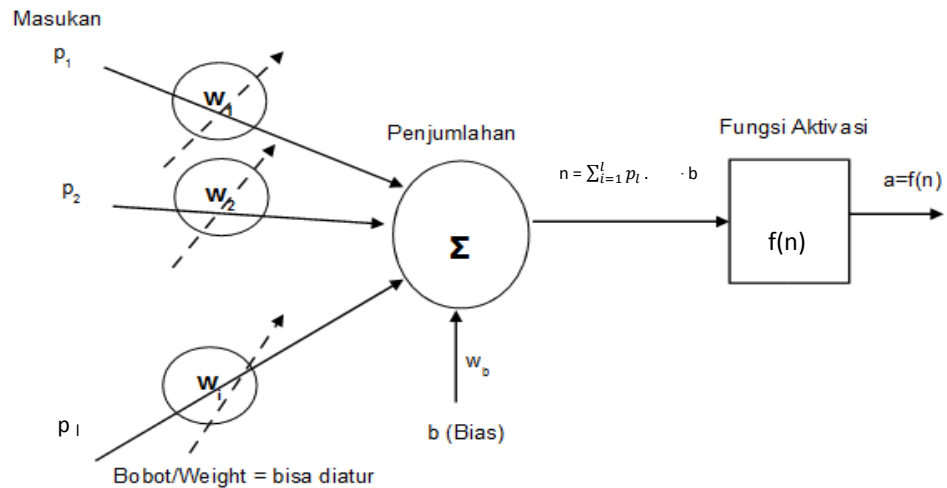
Gambar 2.6. JST Multi *Layer* [6].

### 2.7.2. Algoritma Pembelajaran JST

JST pada dasarnya memiliki dua macam pelatihan yang dilakukan yaitu pelatihan supervisi (*supervised*) dan tanpa supervisi (*unsupervised*). Pelatihan *supervised* yaitu pelatihan yang dilakukan dengan melatih jaringan untuk mendapatkan bobot yang diinginkan sesuai dengan data masukan dan target yang digunakan sebagai acuan untuk menentukan pola jaringan. Sedangkan pelatihan *unsupervised* merupakan pelatihan dengan pengelompokkan input yang serupa dan tidak memerlukan target. Pada pelatihatannya perubahan bobot jaringan dilakukan berdasarkan parameter tertentu dan jaringan dimodifikasi sesuai dengan parameter tersebut [6].

### 2.7.3. Fungsi Aktivasi, Bias dan Threshold

JST ditentukan dengan pola hubungan antar neuron (disebut arsitektur jaringan), metode untuk menentukan bobot penghubung (disebut metode *training/learning/algorithm*) dan fungsi aktivasi. Berikut ini merupakan contoh neuron [6] :



Gambar 2.7. JST Neuron .

Fungsi aktivasi digunakan untuk menentukan keluaran suatu neuron. Argumen fungsi aktifasi adalah net masukan (kombinasi linier masukan dan bobotnya). Jika  $net = \sum p_i \cdot w_i$  maka fungsi aktivasinya adalah  $f(net) = f(\sum p_i \cdot w_i)$ . Pada JST ditambahkan unit masukan yang nilainya selalu = 1 (bias). Bias dapat dipandang sebagai sebuah input yang nilainya = 1. Bias berfungsi untuk mengubah nilai *threshold* menjadi = 0.

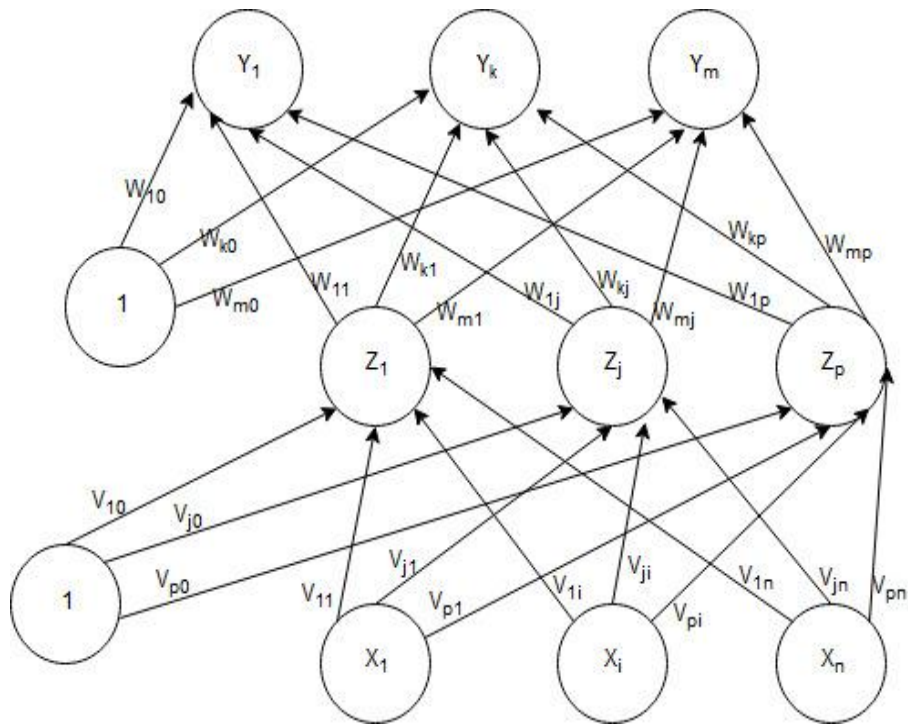
$$net = b + \sum p_i \cdot w_i \dots\dots\dots [6]$$

Neuron – neuron pada JST akan dikumpulkan dalam lapisan yang disebut dengan *layers*. Neuron dalam satu lapisan akan dihubungkan dengan neuron pada lapisan lainnya, terkadang muncul juga *layer* tersembunyi (*hidden layer*) untuk menambahkan keakuratan pelatihan. Neuron terdiri dari 3 elemen pembentuk yaitu himpunan unit-unit yang dihubungkan dengan jalur koneksi dan memiliki bobot yang berbeda-beda, unit penjumlahan yang menjumlahkan input-input sinyal yang sudah dikalikan dengan bobotnya dan yang terakhir yaitu fungsi aktifasinya yang akan menentukan apakah sinyal dari input neuron akan diteruskan ke neuron lain atau tidak.

Kelebihan JST yaitu mampu mengakuisisi pengetahuan kalau tidak ada kepastian (*fault tolerance*), gangguan dapat dianggap sebagai *noise* (derau/gangguan) saja. Kemudian mampu melakukan generalisasi (kesimpulan secara umum pada sebuah kejadian) dan ekstraksi (pemisahan) dari suatu pola data tertentu, JST dapat menciptakan suatu pola pengetahuan melalui pengaturan diri atau kemampuan belajar (*self organizing*). Lalu kemampuan perhitungan secara paralel sehingga proses lebih singkat dan mampu bekerja secara paralel layaknya otak manusia bekerja. Kelemahan JST yaitu kurang mampu untuk melakukan operasi-operasi numerik dengan *presisi* (keadaan pasti) tinggi, keakuratan yang tinggi sulit diperoleh, bekerja berdasarkan pola yang terbentuk pada inputnya, dan lamanya proses *training* yang mungkin terjadi dalam waktu yang sangat lama untuk jumlah data yang besar [6].

## 2.8. **Arsitektur *Backpropagation***

*Backpropagation* merupakan metode dengan melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan untuk mengenali pola yang digunakan selama pelatihan serta kemampuan jaringan untuk memberikan responnya yang benar terhadap pola masukan yang serupa (tapi tidak sama) dengan pola yang dipakai selama pelatihan. *Backpropagation* memiliki beberapa unit yang ada dalam satu atau lebih layar tersembunyi. Berikut ini merupakan gambar arsitektur *backpropagation* [6] :



Gambar 2.8. Arsitektur *Backpropagation* [6].

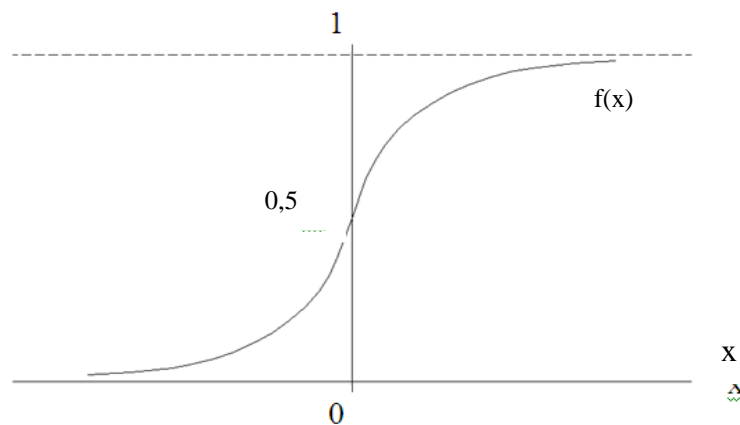
Pada gambar 2.8. merupakan arsitektur *backpropagation* dengan  $n$  unit input ( $X_1, X_2, X_n$ ), sebuah layer tersembunyi ( $Z_1, Z_j, Z_p$ ), dan 3 buah output ( $Y_1, Y_k, Y_m$ ). Dengan urutan pembelajaran pada JST *backpropagation* pertama melakukan inisialisasi bobot, bias dan penetapan *epoch* (satu siklus pelatihan yang melibatkan semua pola), laju pembelajaran (*learning rate*) dan batas toleransi *error*. Selanjutnya dilakukann perhitungan keluaran pada tiap unit tersembunyi ( $Z$ ), kemudian hitung keluaran pada unit keluaran ( $Y$ ) tahapan ini disebut propagasi mau. Tahapan kedua yaitu menghitung *error* pada unit keluaran berdasarkan kesalahan pada unit keluaran ( $Y$ ), selanjutnya hitung *error* pada unit tersembunyi ( $Z$ ) tahapan ini disebut propagasi mundur. Terakhir yaitu tahapan perubahan bobot yaitu dengan menghitung semua perubahan bobot. Proses ini diklakukan secara berurutan dan proses ini akan dilakukan secara berurutan dan berulang kali hingga syarat berhenti terpenuhi.

### 2.8.1. Fungsi Aktivasi *Backpropagation*

Pada *backpropagation* terdapat fungsi aktivasi yang digunakan harus mengikuti beberapa syarat yaitu kontinu, terdiferensial dengan mudah dan merupakan fungsi yang tidak turun. Salah satu fungsi yang memiliki ketiga syarat tersebut sehingga sering dipakai adalah fungsi sigmoid biner yang memiliki range (0,1) [6].

$$f(x) = \frac{1}{1+e^{-x}} \text{ dengan turunan } f'(x) = f(x)(1-f(x)) \dots\dots\dots[6]$$

Grafik fungsinya tampak pada gambar di bawah ini :

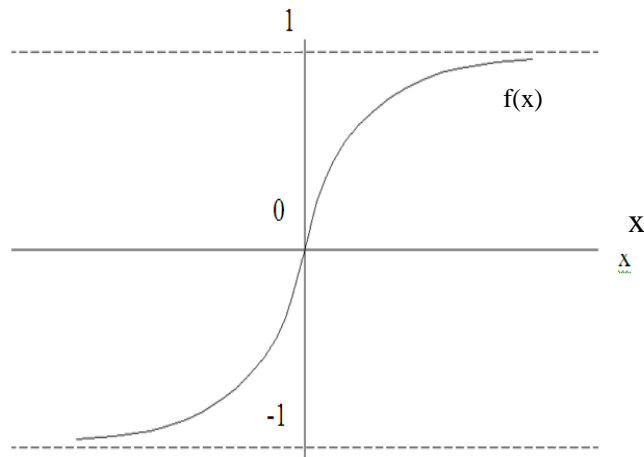


Gambar 2.5.1a. Fungsi Sigmoid Biner [6].

Selain fungsi sigmoid biner terdapat pula fungsi sigmoid bipolar yang memiliki bentuk fungsinya mirip dengan fungsi sigmoid biner, tetapi berbeda range (-1, 1).

$$f(x) = \frac{2}{1+e^{-x}} - 1 \text{ dengan turunan } f'(x) = \frac{(1+f(x))(1-f(x))}{2} \dots [6]$$

Grafik fungsinya tampak pada gambar di bawah ini :



Gambar 2.5.1b. Fungsi Sigmoid Bipolar [6].

### 2.8.2. Pelatihan *Backpropagation*

*Backpropagation* memiliki pelatihan dengan 3 fase. Fase pertama yaitu fase maju dengan pola masukkan dihitung maju mulai dari layar masukan hingga layar keluaran menggunakan fungsi aktivasi yang ditentukan. Fase kedua yaitu fase mundur, selisih antara keluaran jaringan dengan target yang diinginkan merupakan kesalahan yang terjadi. Kesalahan tersebut dipropagasi mundur, dimulai dari garis yang berhubungan langsung dengan unit-unit di layar keluaran. Fase ketiga yaitu modifikasi bobot untuk menurunkan kesalahan yang terjadi [6].

- **Fase I : Propagasi maju**

Pada propagasi maju, sinyal masukan ( $= x_i$ ) dipropagasikan ke layar tersembunyi menggunakan fungsi aktivasi yang ditentukan. Keluaran dari setiap unit layar tersembunyi ( $= z_j$ ) tersebut selanjutnya dipropagasikan maju lagi ke layar tersembunyi di atasnya menggunakan fungsi aktivasi yang ditentukan. Demikian seterusnya hingga menghasilkan keluaran jaringan ( $= y_k$ ). Berikutnya, keluaran jaringan ( $= y_k$ )

dibandingkan dengan target yang harus dicapai ( $= t_k$ ). Selisih  $t_k - y_k$  adalah kesalahan yang terjadi. Jika kesalahan ini lebih kecil dari batas toleransi yang ditentukan, maka iterasi dihentikan. Akan tetapi apabila kesalahan masih lebih besar dari batas toleransinya, maka bobot setiap garis dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi [6].

- **Fase II : Propagasi mundur**

Berdasarkan kesalahan  $t_k - y_k$ , dihitung faktor  $\delta_k$  ( $k = 1, 2, \dots, m$ ) yang dipakai untuk mendistribusikan kesalahan di unit  $y_k$ .  $\delta_k$  juga dipakai untuk mengubah bobot garis yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor  $\delta_j$  di setiap unit di layar tersembunyi sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di layar di bawahnya. Demikian seterusnya hingga semua faktor  $\delta$  di unit tersembunyi yang berhubungan langsung dengan unit masukan dihitung [6].

- **Fase III : Perubahan bobot**

Setelah semua faktor  $\delta$  dihitung, bobot semua garis dimodifikasi bersamaan. Perubahan bobot suatu garis didasarkan atas faktor  $\delta$  neuron di layar atasnya. Sebagai contoh, perubahan bobot garis yang menuju ke layar keluaran didasarkan atas  $\delta_k$  yang ada di unit keluaran [6].

Pada ketiga fase tersebut dilakukan perulangan secara terus menerus hingga kondisi penghentian dipenuhi. Umumnya kondisi penghentian yang sering dipakai adalah jumlah iterasi atau kesalahan. Iterasi akan dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan, atau jika kesalahan yang terjadi sudah lebih kecil dari



batas toleransi yang diizinkan. Berikut ini merupakan tahapan algoritma pelatihan untuk jaringan dengan satu layer tersembunyi (fungsi aktivasi sigmoid biner) adalah sebagai berikut:

**Langkah 0** : Memberikan nilai semua bobot dengan bilangan acak kecil.

**Langkah 1** : Lakukan langkah 2 - 9.

**Langkah 2** : Pada data pelatihan, lakukan langkah 3– 8.

Fase I : Propagasi maju

**Langkah 3** : Tiap unit masukkan menerima sinyal dan meneruskan ke unit tersembunyi di atasnya.

**Langkah 4** : Hitung semua keluaran di unit tersembunyi  $z_j$  ( $j = 1, 2, \dots, p$ ) :

$$z\_net_j = v_{j0} + \sum_{i=1}^n x_i v_{ji} \dots\dots\dots [6].$$

$$z_j = f(z\_net_j) = \frac{1}{1+e^{-z\_net_j}} \dots\dots\dots [6].$$

**Langkah 5** : Hitung keluaran pada jaringan di unit  $y_k$  ( $k=1,2, \dots, m$ ) :

$$y\_net_k = w_{k0} + \sum_{j=1}^p z_j w_{kj} \dots\dots\dots [6].$$

$$y_k = f(y\_net_k) = \frac{1}{1+e^{-y\_net_k}} \dots\dots\dots [6].$$

Fase II : Propagasi mundur

**Langkah 6** : Hitung faktor  $\delta$  unit keluaran berdasarkan kesalahan di setiap unit keluaran  $y_k$  ( $k = 1,2, \dots, m$ )

$$\delta_k = (t_k - y_k) f'(y\_net_k) = (t_k - y_k) y_k (1 - y_k) \dots\dots\dots [6].$$

$\delta_k$  merupakan unit kesalahan yang akan dipakai dalam perubahan bobot layar di bawahnya (langkah 7). Hitung suku perubahan bobot  $w_{kj}$  (yang akan dipakai nanti untuk merubah bobot  $w_{kj}$ ) dengan laju percepatan  $\alpha$ .

$$\Delta w_{kj} = \alpha \delta_k z_j \quad ; \quad k = 1,2, \dots, m ; \quad j = 0,1, \dots, p$$

**Langkah 7** : Hitung faktor  $\delta$  unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi  $z_j$  ( $j = 1, 2, \dots, p$ )

$$\delta_{netj} = \sum_{k=1}^m \delta_k w_{kj} \dots\dots\dots [6].$$

Faktor  $\delta$  unit tersembunyi :

$$\delta_j = \delta_{netj} f'(z_{netj}) = \delta_{netj} z_j (1 - z_j) \dots\dots\dots [6].$$

Hitung suku perubahan bobot  $v_{ji}$  (yang akan dipakai nanti untuk merubah bobot  $v_{ji}$ )

$$\Delta v_{ji} = \alpha \delta_j x_i \quad ; \quad j = 1, 2, \dots, p ; \quad i = 0, 1, \dots, n$$

Fase III : Perubahan Bobot

**Langkah 8** : Hitung semua perubahan bobot

Perubahan bobot garis yang menuju ke unit keluaran :

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) + \Delta w_{kj} \dots\dots\dots [6].$$

Perubahan bobot garis yang menuju ke unit tersembunyi

$$v_{ji}(\text{baru}) = v_{ji}(\text{lama}) + \Delta v_{ji} \dots\dots\dots [6].$$

Kemudian jika telah selesai pelatihan maka jaringan dapat dipakai untuk pengenalan pola. Dalam hal ini, hanya propagasi maju (langkah 4 dan 5) saja yang dipakai untuk menentukan keluaran jaringan. Apabila fungsi aktivasi yang dipakai bukan simoid biner, maka langkah 4 dan 5 harus disesuaikan. Demikian juga turunannya pada langkah 6 dan 7 [6].

## 2.9. Optimalitas Arsitektur *Backpropagation*

Pada *backpropagation* masalah utama yang sering dihadapi yaitu lamanya iterasi yang harus dilakukan. *Backpropagation* tidak dapat memberikan kepastian tentang berapa *epoch* yang harus dilalui untuk mencapai kondisi yang diinginkan [6].

### **2.9.1. Pemilihan Bobot dan Bias Awal**

Pemilihan bobot awal akan mempengaruhi apakah jaringan mencapai titik minimum lokal atau global. Bobot yang menghasilkan nilai turunan aktivasi yang kecil sedapat mungkin dihindari karena akan menyebabkan perubahan bobotnya menjadi sangat kecil. Demikian pula nilai bobot awal tidak boleh terlalu besar karena nilai turunan fungsi aktivasinya menjadi sangat kecil. Sebab itu standar *backpropagation*, bobot dan bias diisi dengan bilangan acak kecil [6].

### **2.9.2. Jumlah Pola Pelatihan**

*Backpropagation* tidak dapat memastikan mengenai kepastian berapa banyak pola yang diperlukan untuk jaringan agar dapat dilatih dengan sempurna. Jumlah pola yang dibutuhkan dipengaruhi oleh banyaknya bobot dalam jaringan serta tingkat akurasi yang diharapkan.

### **2.9.3. Lama Iterasi**

Pada *backpropagation* tujuan utama penggunaan yaitu mendapatkan keseimbangan antara pengenalan pola pelatihan secara benar dan respon yang baik untuk pola lain yang sejenis (data pengujian). Jaringan dapat dilatih terus menerus hingga semua pola pelatihan dikenali dengan benar. Akan tetapi hal itu tidak menjamin jaringan akan mampu mengenali pola pengujian dengan tepat. Pada umumnya data dibagi menjadi 2 bagian saling asing, yaitu pola data yang dipakai sebagai pelatihan dan data yang dipakai untuk pengujian. Perubahan bobot dilakukan

berdasarkan pola pelatihan. Akan tetapi selama pelatihan, kesalahan yang terjadi dihitung berdasarkan semua data (pelatihan dan pengujian). Selama kesalahan ini menurun, pelatihan akan terus dijalankan. Akan tetapi jika kesalahannya meningkat maka pelatihan tidak ada gunanya untuk diteruskan lagi [6].

#### **2.9.4. Momentum**

Pada metode *backpropagation* momentum digunakan untuk mengurangi perubahan bobot yang mencolok atau data yang sangat berbeda dengan yang lainnya. Momentum merupakan perubahan bobot yang didasarkan pada gradien pada pola terakhir dan pola sebelumnya atau pola terakhir saja yang digunakan [6].

### **2.10. *Backpropagation* dengan MATLAB**

Matlab dibuat untuk memudahkan perhitungan tersebut. Matlab juga menyediakan fungsi-fungsi khusus untuk menyelesaikan model jaringan syaraf tiruan metode *backpropagation*.

#### **2.10.1. Matriks dan Vektor**

Pada proses JST banyak berhubungan dengan matriks dan vektor untuk itu perlu memahami mengenai matriks dan vektor. Matriks merupakan kumpulan bilangan-bilangan yang disusun dalam larik baris dan kolom. Umumnya matriks diberi notasi huruf kapital A,B,...Matriks A yang terdiri dari  $m$  baris dan  $n$  kolom (sering disebut dengan ordo  $m \times n$ ), maka akan ditulis sebagai :

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{1n} \\ a_{21} & a_{22} & a_{2n} \\ a_{m1} & a_{m2} & a_{mn} \end{bmatrix}$$

Matriks yang terdiri dari 1 kolom sama dengan suatu vektor. Vektor merupakan ruang dimensi n dengan bilangan-bilangan riil. Notasinya dengan menggunakan huruf-huruf kecil seperti x, y, z ... . Pada vektor tidaklah penting dinyatakan dalam sebuah baris ataupun sebuah kolom.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_n \end{bmatrix} \quad \text{dengan } x_1, x_2, \dots, x_n \text{ adalah bilangan – bilangan riil.}$$

## 2.11. Penelitian Terkait

Identifikasi Kematangan Buah Tomat Menggunakan Metode *Backpropagation* oleh Desa Deswari, Hendrick MT, Derisma MT, 2013.

Penelitian ini bertujuan mengidentifikasi tingkat kematangan buah tomat berdasarkan warna dengan jaringan syaraf tiruan *backpropagation* untuk menghasilkan nilai bobot. Nilai bobot yang dihasilkan digunakan untuk identifikasi buah tomat masak, muda, dan setengah masak [14].

Sistem Otomasi Dalam Penyortiran Tomat Dengan Image Processing Menggunakan Metode Deteksi RGB oleh Ihsan Nugraha P M, Drs. Suwandi M.Si, Hertiana Bethaningtyas S.T, M.T , 2015.

Penelitian ini bertujuan mengidentifikasi warna tomat yang matang secara tampilan visual didapat dari proses mendeteksi nilai RGB (Red, Green dan Blue) pada warna tomat. Identifikasi tekstur warna pada tomat dengan ekstraksi cirinya dengan menggunakan metode deteksi warna dengan informasi yang didapatkan berupa nilai-nilai ciri statistik. Ekstraksi ciri citra diakuisisi dari kamera kemudian diklasifikasi dengan *euclidean distance* [20].

Pemrosesan Citra Digital Untuk Klasifikasi Mutu Buah Pisang Menggunakan Jaringan Syaraf Tiruan. 2014.

Penelitian ini bertujuan mengolah citra pisang dengan pemrosesan citra digital untuk mengekstrak fitur warna dan tekstur buah pisang. Klasifikasi mutu pisang menggunakan jaringan syaraf tiruan *backpropagation* dengan parameter luas cacat, nilai red, green, blue, energy, homogeneity, dan contrast. Sistem klasifikasi mutu pisang dengan laju pembelajaran 0,3 dan jumlah neuron pada lapisan tersembunyi sebanyak 10 neuron dengan tingkat keberhasilan 94% dari 100% data uji pisang [21].

Peningkatan Kontras Menggunakan Metode *Contrast Limited Adaptive Histogram Equalization* Pada Citra *Underwater*. 2015.

Penelitian ini bertujuan mengubah kualitas citra *underwater* yang rendah karena cahaya matahari sulit menembus kedalaman air metode CLAHE (*Contrast Limited Adaptive Histogram Equalization*) metode yang mampu meningkatkan kontras citra tetapi tidak berlebihan. Metode CLAHE beroperasi pada tile, kontras yang terdapat pada tiap – tiap tile akan diperbaiki dan mendapatkan hasil yang sesuai dengan yang diinginkan. Hasil penerapan metode CLAHE terbaik akan ditentukan dengan nilai MSE dan PSNR tertinggi [22].

## BAB III

### METODOLOGI PENELITIAN

### 3.1. Waktu dan Tempat

Penelitian dan pembuatan Tugas Akhir ini dilakukan pada :

Waktu : Juni 2018 – Desember 2018.

Tempat : Kebun kopi Sinarwaya Kab. Pringsewu Lampung

Jadwal kegiatan penelitian yang dilakukan dapat dilihat pada Tabel 3.1 dibawah ini :

Tabel 3.1 Jadwal Kegiatan Penelitian

[illegible]

### 3.2. Alat dan Bahan

Berikut ini merupakan alat yang digunakan pada penelitian dan pembuatan Tugas Akhir ini adalah :

#### 1. Alat Penelitian

##### a. Perangkat Keras

- ✓ Laptop Dell latitude 6320 dengan spesifikasi *Processor* Intel(R) Core(TM) i7-2620M CPU @270GHz, RAM 4GB, System 64-bit.
- ✓ Kamera handphone Xiomi Note 4 sengan spesifikasi kamera 13MP.




##### b. Perangkat Lunak

- ✓ Sistem operasi Windows 7.
- ✓ MATLAB r2015a, digunakan untuk pembuatan sistem.
- ✓ Paint, digunakan untuk mengubah ukuran citra (*scaling*).
- ✓ Power Designer, digunakan untuk pembuatan *flowchart*.













#### 2. Bahan Penelitian






Buah Kopi yang dipetik langsung dari kebun kopi yang berada di sinarwaya pringsewu lampung dengan 35 gambar Sample Latih berikut ini :

Tabel 3.1. Sample Latih buah kopi untuk pelatihan







		
Sample Latih 1	Sample Latih 2	Sample Latih 3



		
<b>Sample Latih 4</b>	<b>Sample Latih 5</b>	<b>Sample Latih 6</b>
		
<b>Sample Latih 7</b>	<b>Sample Latih 8</b>	<b>Sample Latih 9</b>
		
<b>Sample Latih 10</b>	<b>Sample Latih 11</b>	<b>Sample Latih 12</b>
		
<b>Sample Latih 13</b>	<b>Sample Latih 14</b>	<b>Sample Latih 15</b>

 <p><b>Sample Latih 16</b></p>	 <p><b>Sample Latih 17</b></p>	 <p><b>Sample Latih 18</b></p>
 <p><b>Sample Latih 19</b></p>	 <p><b>Sample Latih 20</b></p>	

Tabel 3.2. Sample Uji buah kopi untuk pelatihan

 <p><b>Sample Uji 1</b></p>	 <p><b>Sample Uji 2</b></p>	 <p><b>Sample Uji 3</b></p>
 <p><b>Sample Uji 4</b></p>	 <p><b>Sample Uji 5</b></p>	 <p><b>Sample Uji 6</b></p>



**Sample Uji 7**



**Sample Uji 8**



**Sample Uji 9**



**Sample Uji 10**



**Sample Uji 11**



**Sample Uji 12**



**Sample Uji 13**



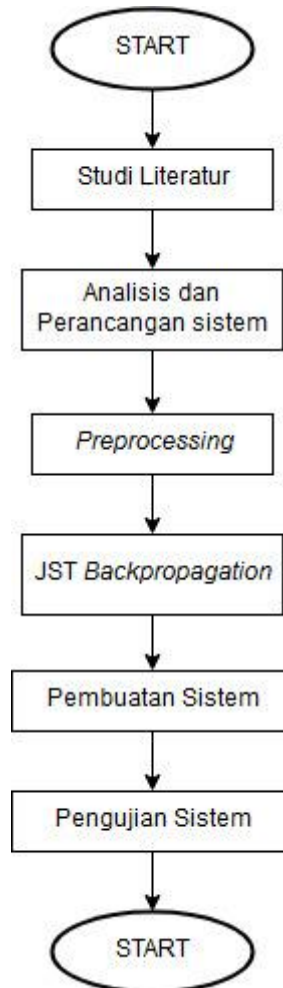
**Sample Uji 14**



**Sample Uji 15**

### 3.3. Tahapan Penelitian

Pada penelitian ini tahapan penelitian atau langkah-langkah yang dilakukan yaitu sebagai berikut :



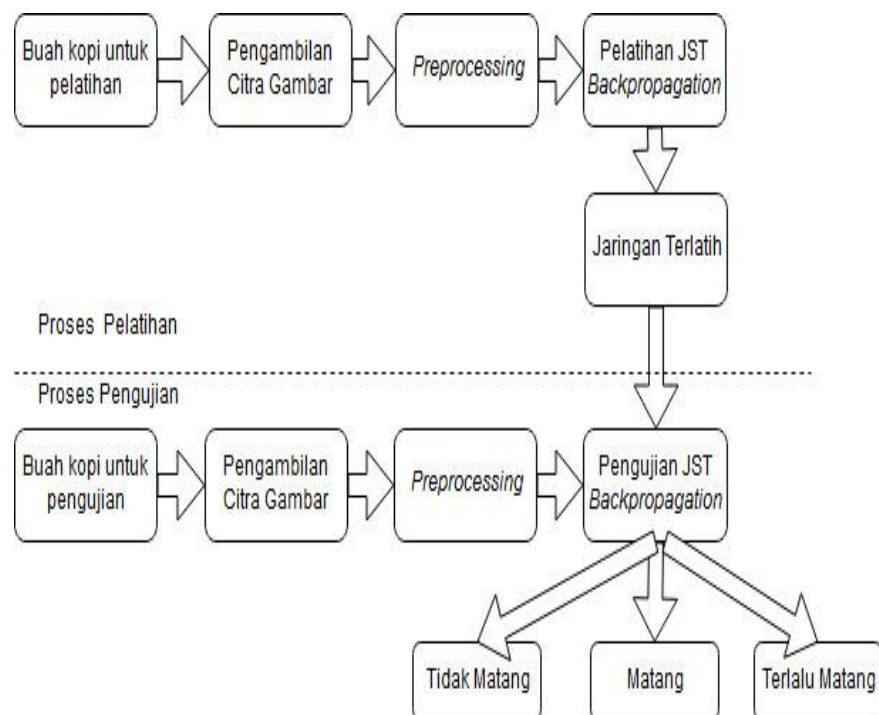
Gambar 3.1. Tahapan Penelitian

#### 3.3.1. Analisis dan Perancangan Sistem

Analisis yang dilakukan pada penelitian ini dengan penjabaran penyusunan sistem dan mengidentifikasi masalah dengan menentukan nilai masukan, nilai keluaran, arsitektur jaringan, nilai ambang MSE (*Mean Squared Error*) sebagai kondisi berhenti,



*learning rate*, goal dan menentukan momentum. Penentuan nilai bobot ditentukan secara acak pada interval misalnya  $[-1, +1]$  atau  $[0.5, +0.5]$  ataupun lainnya (tidak adanya aturan baku mengenai interval ini). Pada perancangan sistem ini dilakukan pengolahan citra digital dengan ekstraksi ciri citra warna RGB (*preprocessing*), kemudian dilakukan pemrosesan dengan JST backpropagation yang terdiri dari 2 proses, yaitu proses *training* (pelatihan) dan proses *testing* (pengujian). Pada proses pelatihan dilakukan untuk melatih jaringan syaraf tiruan agar mampu mengenali dan mengidentifikasi warna buah kopi yang matang keseluruhan saat proses sortasi. Buah kopi sendiri disortasi untuk meningkatkan mutu dan kualitas kopi yang dihasilkan dengan mengklasifikasi 3 output keluaran yaitu tidak matang, matang dan terlalu matang. Berikut ini merupakan diagram alir untuk cara kerja dari sistem untuk memperjelas langkah-langkah kerja yang akan dilakukan pada sistem:



Gambar 3.2. Perancangan Sistem Sortasi Buah Kopi

### **3.3.2. Tahap *Preprocessing***

#### **3.3.2.1. Pengambilan Citra Gambar Buah Kopi**

Pada proses ini dilakukan pengambilan citra buah kopi menggunakan kamera handphone Xiaomi Note 4 dengan metode pengambilan sampel *random* (acak) yaitu berwarna merah tua, merah, merah muda, hijau kekuningan dan hijau. Citra yang diambil terdiri dari 5 citra warna dengan posisi pengambilan acak dan terdiri dari lebih dari satu buah kopi dengan format JPEG (*Joint Photographic Experts Group*). Kemudian citra disimpan dalam folder dengan nama Sample

#### **3.3.2.2. *Scaling***

Pada proses ini *scaling* atau penskalaan merupakan suatu proses mengubah ukuran citra dengan tujuan untuk menyamakan seluruh ukuran citra. Ukuran citra pada data latih dan data uji akan menjadi sama yaitu dengan ukuran 500 x 500 piksel dengan tujuan agar citra tidak memiliki banyak nilai untuk diolah pada proses klasifikasi. Pada proses ini dilakukan dengan cara manual yaitu menggunakan aplikasi *Microsoft Paint* dengan melakukan *resize* ukuran pixel gambar yang kemudian file disimpan.

### 3.3.2.3. *Contrast- Limited Adaptive Histogram Equalization (CLAHE)*

*Image Enhancement* (Perbaikan Citra) dengan algoritma CLAHE membagi citra menjadi beberapa daerah kontekstual dan menerapkan pemerataan histogram dari masing – masing daerah tersebut. Metode ini akan meratakan distribusi nilai warna yang digunakan sehingga ciri dan kontras citra bisa terlihat, terutama pada daerah yang homogen dan menghindari penguatan dan gangguan (noise) yang ada pada citra. Pada penggunaannya pada Matlab menggunakan perintah *Adapthisteq*.

### 3.3.2.4. *Grayscale*

Pada proses ini citra akan melalui tahap *grayscale* yang merupakan perubahan citra warna menjadi citra keabuan setelah melalui proses *scaling*. Untuk mengubah citra berwarna menjadi *grayscale* yaitu dengan cara mengkonversi nilai matrik dengan mengambil rata-rata dari nilai r, g dan b seperti pada rumus di bawah ini :

$$Grayscale = \frac{r+g+b}{3} \dots\dots\dots ( 3.1 )$$

Keterangan :

- r*        = Nilai *Red*
- g*        = Nilai *Green*
- b*        = Nilai *Blue*

### 3.3.2.5. Thresholding

Proses *thresholding* merupakan tahapan pengubahan kuantitas pada citra atau pengubahan citra menjadi citra biner atau hitam putih. *Thresholding* pada proses ini digunakan untuk mengatur jumlah derajat keabuan yang ada pada citra. Menggunakan *thresholding* dilakukan dengan langkah berikut ini :

- Nilai *threshold* ( $T$ ) ditentukan dengan rentan 0 -255, pada penelitian ini diambil nilai  $T = 165$ .
- Nilai piksel jika didapat lebih dari atau sama dengan 165 maka nilai piksel pada citra diubah menjadi 1, sedangkan nilai piksel yang didapat kurang dari 138 maka diubah menjadi 0.

Berikut ini merupakan persamaan yang digunakan untuk mengubah citra grayscale menjadi biner dengan metode *thresholding* yaitu :

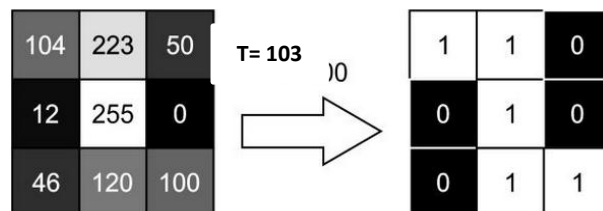
$$g(x,y) = \begin{cases} 1, & \text{jika } f(x,y) \geq T \\ 0, & \text{jika } f(x,y) < T \end{cases} \dots\dots\dots (3.2)$$

Keterangan :

$f(x,y)$  : Nilai citra grayscale

$y(x,y)$  : Nilai citra biner

$T$  : Nilai *thresholding*



Gambar 3.3. proses perubahan nilai pixel pada proses Thresholding



### 3.3.3. Jaringan Syaraf Tiruan *Backpropagation*

Pada proses ini JST *backpropagation* mula-mula memisahkan data untuk pelatihan dan pengujian (*training* dan *testing*) jaringan. Umumnya *data* set terbagi menjadi dua yaitu pelatihan dan pengujian. Pelatihan dilakukan untuk menentukan bobot dan untuk mengklasifikasikan data untuk membentuk suatu pola. Selanjutnya *mechine learning* akan mempelajari pola dan akan disimpan. Kemudian dilakukan pengujian untuk melihat tingkat keberhasilan yang dilakukan sistem, untuk itu data yang digunakan pada proses *training* dan *testing* harus berbeda sehingga dapat diketahui apakah pola yang dibaca oleh sistem sudah dapat membaca dengan benar atau tidak.

Pada proses pelatihan JST bertujuan untuk menentukan nilai bobot yang menghubungkan antara input ke *hidden layer* ke output dan meminimkan kesalahan yang terjadi dan mengidentifikasi apakah model JST *backpropagation* ini sudah sesuai dengan target yang diharapkan. Pelatihan ini memerlukan data input dan data target. Pada pelatihan JST *backpropagation* terdapat dua cara umum pelatihan yaitu *backpropagation* dan *backpropagation* dengan momentum. Berikut ini merupakan persamaan dengan cara *backpropagation* momentum :

$$w_{kj}(t+1) = w_{kj}(t) + \alpha \delta_k z_j + \mu(w_{kj}(t) - w_{kj}(t-1)) \dots\dots\dots [6].$$

Perubahan bobot menuju ke unit tersembunyi

$$v_{ji}(t+1) = v_{ji}(t) + \alpha \delta_k z_j + \mu(v_{ji}(t) - v_{ji}(t-1)) \dots\dots\dots [6].$$

- $\mu$  merupakan momentum yang nilainya antara 0 dan 1.
- $w_{kj}(t+1) / v_{ji}(t+1)$  merupakan perubahan bobot dengan momentum.
- $w_{kj}(t-1) / v_{ji}(t-1)$  merupakan bobot mula-mula pada iterasi pola pertama (bobot awal).

Pelatihan JST *backpropagation* memiliki dua hal penting yaitu pada *learning rate* dan *momentum*. Fungsi *learning rate* yaitu untuk menentukan seberapa cepat jaringan tersebut dapat mempelajari pola dari data pelatihan. Fungsi *momentum* yaitu untuk meningkatkan kecepatan menemukan nilai yang diharapkan. Penambahan *momentum* digunakan untuk menghindari bobot yang terlalu mencolok akibat adanya data yang sangat berbeda. Penentuan nilai keduanya harus ditentukan dengan benar untuk meminimalkan proses pembelajaran yang terlalu lama jika nilai terlalu kecil dan akan terjadi kesalahan pembacaan pola jika nilai terlalu besar. Momentum menunjukkan bahwa penentuan bobot pada iterasi sebelumnya mempengaruhi bobot saat ini.

#### **3.3.3.1. Arsitektur JST *Backpropagation***

Pada penelitian ini arsitektur JST *backpropagation* yang digunakan untuk *training* terdiri dari 20 sample gambar yang di ambil secara *random* dengan isi buah kopi lebih dari 1 buah kopi. Pada penelitian ini dibuat parameter-parameter jaringan sehingga dapat menghasilkan output yang sesuai dengan target yang diharapkan. Berikut ini merupakan parameter yang ditentukan untuk menghasilkan output yang optimal dan sesuai dengan target yang diharapkan :

- Parameter jumlah iterasi (*epoch*)
- Parameter untuk batas minimal nilai *error* atau mse.
- Parameter *learning rate* untuk menentukan seberapa cepat jaringan dapat mengenali pola.
- Parameter *momentum* untuk kecepatan menemukan nilai yang diharapkan.

### **3.3.3.2. Fungsi Aktivasi**

Menurut Siang (2005), pada *backpropagation* fungsi aktivasi harus memenuhi beberapa syarat dalam penentuannya yaitu bersifat kontinu, terdiferensial dengan mudah dan bukan merupakan fungsi turun [6]. Fungsi aktivasi yang digunakan pada penelitian ini dan memenuhi ketiga syarat tersebut yaitu fungsi *sigmoid biner* yang memiliki range (0,1).

### **3.3.3.3. Pemilihan Bobot**

Pada penentuan bobot pada penelitian ini dilakukan secara acak dan terdiri dari bilangan acak kecil, karena apabila bobot menghasilkan nilai turunan aktivasi kecil dapat menyebabkan perubahan bobot menjadi sangat kecil. Demikian apabila nilai bobot awal terlalu besar maka nilai turunan fungsi aktivasi menjadi sangat kecil pula.

### **3.3.3.4. Jumlah Unit Tersembunyi**

Penentuan jumlah *hidden layer* dapat ditentukan dengan kebutuhan yang diinginkan. Namun pada dasarnya jaringan dengan sebuah layer tersembunyi sudah cukup untuk JST *backpropagation* untuk mengenali suatu pola antara masukan dan target. Akan tetapi, penambahan *hidden layer* kadangkala membuat pelatihan lebih mudah. Pada penelitian ini terdiri dari 3 buah input dan 3 buah output. Penentuan jumlah *hidden layer* dan jumlah neuron pada tiap *hidden layer* ditentukan pada saat pelatihan.

#### 3.3.3.5. Iterasi

Penggunaan JST *backpropagation* bertujuan untuk mendapatkan keseimbangan antara pengenalan pola pelatihan secara benar. Jaringan dapat dilakukan pelatihan secara terus menerus hingga pola dapat dikenali dengan benar. Namun hal tersebut tidak menjamin jaringan mampu mengenali pola pengujian dengan tepat. Menurut Siang (2005), Melakukan iterasi secara terus menerus tidaklah bermanfaat hingga semua kesalahan pola pelatihan bernilai 0. Kesalahan yang terjadi pada semua data (pelatihan dan pengujian) apabila kesalahan menurun pelatihan dapat terus dijalankan, namun jika kesalahan terus meningkat pelatihan tidak ada gunanya untuk diteruskan karena jaringan telah mengambil sifat yang hanya dimiliki secara spesifik oleh data pelatihan (tidak dimiliki oleh data pengujian) dan mulai kehilangan kemampuan untuk *generalisasi* (menyamarkan) [6].

#### 3.3.3.6. Penentuan *Goal*, *Learning Rate* dan *Momentum*

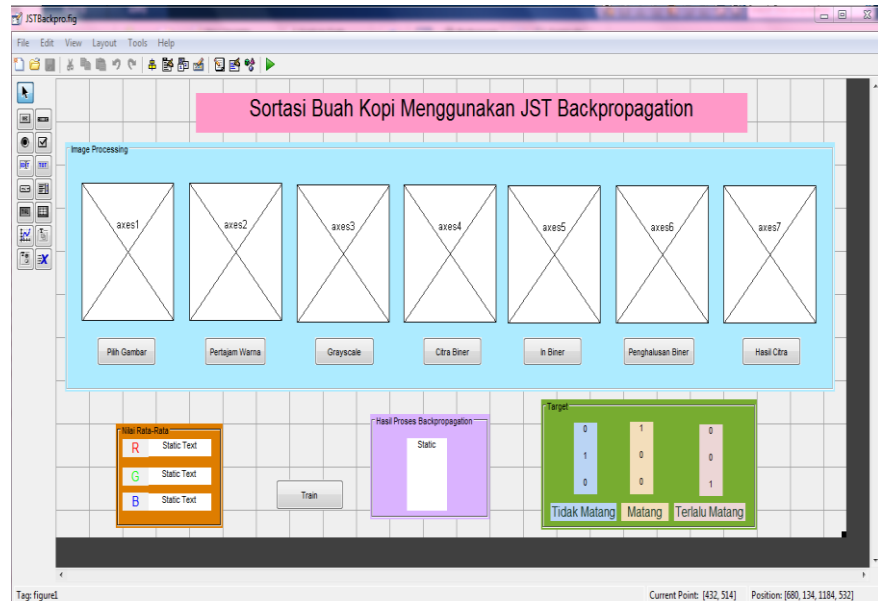
Pada proses belajar jaringan penentuan batasan nilai *error* penting karena dapat membatasi iterasi mse yang dilakukan. Iterasi akan berhenti jika  $mse < \text{batas}$  yang ditentukan yaitu 0.0001. Parameter ini cukup untuk membatasi nilai mse pada pembelajaran dan pelatihan pada penelitian ini. Pada penelitian ini digunakan nilai sebesar 0.01 sebagai nilai *learning rate*. Digunakan nilai tersebut karena *defaultnya* (standar) yang digunakan pada Matlab. Parameter penentuan nilai momentum yaitu berdasarkan *defaultnya* yaitu 0.9.

**Tabel 3.3.** Arsitektur Penelitian

Karakteristik	Spesifikasi
- Jumlah <i>hidden layer</i>	2
- Neuron lapisan input	3
- Neuron lapisan tersembunyi 1	40
- Neuron lapisan tersembunyi 2	15
- Neuron lapisan output	3 (Red, Green, Blue)
- Fungsi aktivasi lapisan <i>hidden layer</i>	<i>Sigmoid Biner</i>
- Fungsi aktivasi lapisan output	<i>Sigmoid Biner</i>
- Parameter <i>epoch</i>	1.000
- Parameter <i>goal</i>	0,0001
- Parameter <i>learning rate</i>	0,01
- Parameter <i>momentum</i>	0,9
- Fungsi pelatihan jaringan	<i>Trainrp</i>
- Fungsi perubahan bobot	<i>Learngdm</i>
- Fungsi perhitungan error	<i>mse</i>

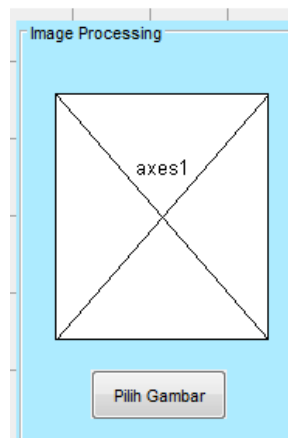
#### 3.3.4. Pembuatan Sistem

Pada penelitian ini sistem dibangun dengan menggunakan aplikasi MATLAB R2015a. Alasan menggunakan aplikasi Matlab karena Matlab dianggap lebih fleksibel dalam pembuatan sistem dikarenakan Matlab memiliki fungsi-fungsi JST terutama JST *backpropagation* [6]. Keunggulan yang dimiliki ini sehingga membuat penulis memilih aplikasi ini sebagai dasar pembuatan sistem dalam penelitian ini. Sistem yang akan dibangun dirancang dengan menggunakan *Grapical User Interface* (GUI) untuk memudahkan penggunaan sistem. Berikut ini merupakan rancangan GUI yang telah dibangun menggunakan Matlab.



Gambar 3.. Pembuatan Sistem GUI

Pada gambar 3.4. merupakan rancangan GUI yang akan ditampilkan pada saat program dijalankan. Program ini terdiri pengolahan citra digital (*preprocessing*) dengan 7 langkah pengolahan yaitu pemilihan citra gambar yang akan diolah, pertajam warna (CLAHE), perubahan warna menjadi *grayscale*, mengubah citra menjadi citra biner (*black and white*), merubah citra menjadi citra *in biner* (membalikkan citra dari 0 menjadi 1 dan sebaliknya), penghalusan citra in biner, dan terakhir penggabungan dari keseluruhan proses yaitu citra hasil yang berfungsi untuk mengambil nilai warna *red*, *green*, dan *blue* yang telah dipertajam warnanya dan dihilangkan *backgroundnya*. Pembuatan GUI pada gambar berguna sebagai *User Interface* untuk memudahkan pengguna dalam melakukan sortasi buah kopi. Pada gambar tersebut terdapat 7 axes yang setiap axesnya berfungsi untuk menampilkan gambar yang akan dilakukan pengolahan citra. Pada Axes 1 akan menampilkan gambar yang dipilih dengan perintah tombol “Pilih Gambar”.



Gambar 3.5. GUI Axes1 dengan perintah Pilih Gambar

Pada gambar 3.5 berfungsi untuk menampilkan citra gambar yang akan diolah pada proses pengolahan citra selanjutnya dengan code program berikut ini .

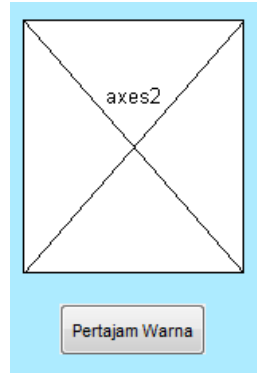
```

129 % --- Executes on button press in pushbutton1.
130 function pushbutton1_Callback(hObject, eventdata, handles)
131 % hObject    handle to pushbutton1 (see GCBO)
132 % eventdata  reserved - to be defined in a future version of MATLAB
133 % handles    structure with handles and user data (see GUIDATA)
134
135 proyek=guidata(gcbo);
136 [namafile,direktori]=uigetfile({'*.jpg'; '*.bmp'; '*.png'; '*.tif'}, 'Buka Gambar')
137 if isequal(namafile,0)
138     return;
139 end
140 eval(['cd '' direktori ''']);
141 I=imread(namafile);
142 set(proyek.figure1, 'CurrentAxes', proyek.axes1);
143 set(imshow(I));
144
145 set(proyek.figure1, 'Userdata', I);
146 set(proyek.axes1, 'Userdata', I);
147

```

Gambar 3.6. Sourcode untuk fungsi pilih gambar.

Pada gambar 3.6 merupakan source code yang digunakan dengan perintah membuka direktori file, kemudian dilakukan pemilihan gambar yang diinginkan pada direktori dengan format gambar .jpeg, .bmp, .tif. kemudian sistem akan menampilkan gambar pada Axes1 yang selanjutnya gambar akan disimpan pada program dengan nama I untuk memudahkan pemanggilan gambar pada proses selanjutnya. Selanjutnya pada Axes2 dengan push button “Pertajam Warna”.



Gambar 3.7. GUI Axes2 dengan perintah pertajam warna.

Pada gambar 3.7. merupakan tampilan GUI dengan perintah perbaikan kontras warna pada gambar dengan menggunakan metode CLAHE. Fungsi ini sendiri digunakan pada pengolahan citra untuk memperbaiki kontras warna citra namun tidak dengan berlebihan sehingga memudahkan dalam proses klasifikasi saat pembacaan nilai RGB pada citra. Berikut ini merupakan source code yang digunakan.

```

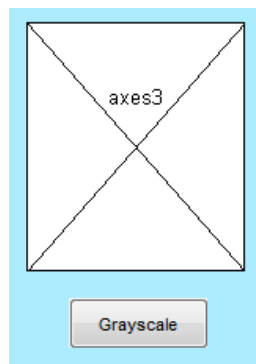
149 % --- Executes on button press in pushbutton2.
150 function pushbutton2_Callback(hObject, eventdata, handles)
151 % hObject    handle to pushbutton2 (see GCBO)
152 % eventdata  reserved - to be defined in a future version of MATLAB
153 % handles    structure with handles and user data (see GUIDATA)
154 proyek=guidata (gcbo);
155 I=get(proyek.axes1,'Userdata');
156
157 Red= I(:, :,1);
158 Green= I(:, :,2);
159 Blue= I(:, :,3); %memisahkan setiap warna RGB
160
161 reds= adapthisteq(Red);
162 greens= adapthisteq(Green);
163 blues= adapthisteq(Blue);
164 %mempertajam kontrasnya dengan Contrast Limited Adaptive Histogram
165 %Equalization (CLAHE) pada tiap layar warnanya
166
167 edgeSharp(:, :,1)= reds;
168 edgeSharp(:, :,2)= greens;
169 edgeSharp(:, :,3)= blues; %menggabungkan kembali ketiga warna
168 edgeSharp(:, :,2)= greens;
169 edgeSharp(:, :,3)= blues; %menggabungkan kembali ketiga warna
170
171 set(proyek.figure1, 'CurrentAxes',proyek.axes2);
172 set(imshow(edgeSharp));
173 set(proyek.figure1, 'Userdata', edgeSharp);
174 set(proyek.axes2, 'Userdata', edgeSharp);
175

```

Gambar 3.8. Source code untuk fungsi pertajam warna.



Pada gambar 3.8. merupakan source code pertajam warna citra dengan menggunakan metode CLAHE pada gambar diatas dapat dilihat pada line 155 merupakan perintah untuk memanggil data pada axes1 yang sebelumnya telah disimpan dengan nama I. Selanjutnya dilakukan pemisahan warna Red, Green , dan Blue pada citra dengan perintah pada line 157-159, setelah dilakukan pemisahan warna RGB selanjutnya digunakan metode CLAHE dengan source code pada Matlab yaitu *adapthisteq*, fungsi ini digunakan untuk mempertajam kontras setiap warna pada RGB citra. Kemudian setelah setiap warna RGB dipertajam dengan metode CLAHE perintah selanjutnya yaitu menggabungkan kembali ketiga warna tersebut dan ditampilkan pada axes2 dengan nama file yang disimpan yaitu edgeSharp. Berikut ini gambar GUI untuk perintah Grayscale.



Gambar 3.9. GUI Axes3 dengan perintah Grayscale

Pada gambar 3.9. merupakan tampilan GUI untuk menampilkan citra grayscale. Fungsi ini sendiri digunakan pada untuk merubah warna citra RGB menjadi warna abu-abu sehingga akan lebih mudah untuk proses selanjutnya yaitu proses threshold. Berikut ini merupakan source code yang digunakan pada proses ini.

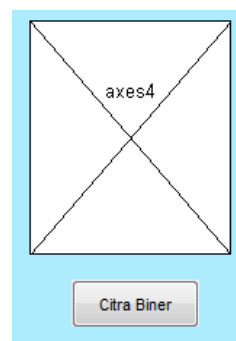
```

177 % --- Executes on button press in pushbutton3.
178 function pushbutton3_Callback(hObject, eventdata, handles)
179 % hObject    handle to pushbutton3 (see GCBO)
180 % eventdata  reserved - to be defined in a future version of MATLAB
181 % handles    structure with handles and user data (see GUIDATA)
182
183 - proyek=guidata (gcbo);
184 - edgeSharp=get(proyek.axes2,'Userdata');
185 - Igray=rgb2gray(edgeSharp);
186 - set(proyek.figure1,'CurrentAxes',proyek.axes3);
187 - set(imshow(Igray));
188 - set(proyek.figure1,'Userdata',Igray);
189 - set(proyek.axes3,'Userdata',Igray);
190

```

Gambar 3.10. Source code untuk perintah Grayscale.

Pada gambar diatas dapat dilihat bahwa sistem menggunakan perintah untuk memanggil data yang sebelumnya disimpan yaitu edgeSharp yang sebelumnya telah ditampilkan pada axes2 fungsi itu terdapat pada line 184. Kemudian pada line 185 dapat dilihat digunakan fungsi rgb2gray fungsi ini merupakan salah satu fungsi otomatis yang dimiliki Matlab untuk mengubah citra warna RGB menjadi citra Grayscale atau abu-abu. Kemudian sistem menyimpan file grayscale ini dengan nama Igray yang memudahkan saat pemanggilan data pada proses pengolahan citra selanjutnya. Berikut ini merupakan tampilan GUI dengan push button citra biner.



Gambar 3.11. GUI axes4 dengan perintah citra biner.

Pada 3.11. merupakan tampilan GUI dengan menampilkan citra biner yang berfungsi untuk mengubah citra yang mana piksel-piksel citra diubah dan hanya memiliki dua buah nilai derajat keabuan (grayscale) yaitu hitam dan putih atau pixel-pixel citra akan bernilai

1 pada objek dan bernilai 0 pada latar belakang. Fungsi ini digunakan untuk memudahkan dalam pembacaan nilai RGB citra dengan cara memisahkan objek gambar dengan *background*. Berikut ini source code yang digunakan pada perintah ini.

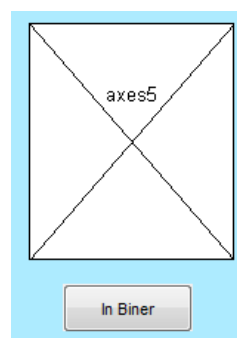
```

198 - proyek=guidata(gcbo);
199 - Igray=get(proyek.axes3,'Userdata');
200 - Ibw = im2bw(Igray,165/255); %mengkonversikan citra gray menjadi citra biner
201 - %dengan threshold
202 - set(proyek.figure1,'CurrentAxes',proyek.axes4);
203 - set(imshow(Ibw));
204 - set(proyek.figure1,'Userdata',Ibw);
205 - set(proyek.axes4,'Userdata',Ibw);
206

```

Gambar 3.12. Source code untuk perintah citra biner.

Pada gambar 3.12. merupakan source code untuk mengubah citra grayscale menjadi citra biner. Pada line 199 sistem memanggil file sebelumnya yaitu Igray yang kemudian diolah gambarnya dengan fungsi im2bw dengan threshold. Pada line 200 dapat dilihat Ibw = im2bw(Igray, 165/255) merupakan fungsi yang digunakan untuk mengkonversi citra grayscale menjadi citra biner dengan threshold citra keabuan bernilai 165/255. Fungsi dari nilai 165 merupakan perubahan nilai piksel  $\geq 165$  pada proses thresholding akan bernilai 1 atau putih pada citra biner. Sedangkan nilai yang kurang dari 165 akan bernilai 0 atau berwarna hitam. Berikut ini merupakan gambar axes4 dengan perintah in biner.



Gambar 3.13.. GUI axes5 dengan perintah in biner

Pada gambar 4.10 merupakan tampilan GUI untuk menampilkan in biner atau perintah untuk membalikkan nilai biner dengan tujuan

untuk menampilkan objek yang akan digunakan dengan membalik nilai biner sebelumnya bernilai 1 menjadi 0 dan nilai 0 menjadi 1. Berikut ini source code yang digunakan pada push button tersebut.

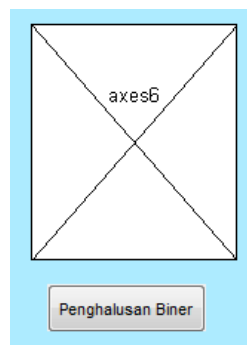
```

208 % --- Executes on button press in pushbutton5.
209 function pushbutton5_Callback(hObject, eventdata, handles)
210 % hObject    handle to pushbutton5 (see GCBO)
211 % eventdata  reserved - to be defined in a future version of MATLAB
212 % handles    structure with handles and user data (see GUIDATA)
213
214 proyek=guidata(gcbo);
215 Ibw=get(proyek.axes4,'Userdata');
216 Ibw = imcomplement(Ibw);
217 set(proyek.figure1,'CurrentAxes',proyek.axes5);
218 set(imshow(Ibw));
219 set(proyek.figure1,'Userdata',Ibw);
220 set(proyek.axes5,'Userdata',Ibw);
221

```

Gambar 3.14.. Source code untuk perintah in biner

Pada gambar 3.15. sama halnya dengan proses sebelumnya yaitu sistem membaca proses sebelumnya yaitu Ibw atau citra biner yang telah di threshold pada axes4 kemudian pada line 216 digunakan fungsi imcomplement (Ibw) yang berfungsi untuk membalikkan nilai biner sehingga tampilan yang akan dimunculkan pada axes5 merupakan kebalikan dari axes4 yaitu background yang semula berwarna putih menjadi hitam dan objek yang semula berwarna hitam berubah menjadi warna putih. Berikut ini merupakan perintah untuk menampilkan proses selanjutnya pada sistem ini yaitu penghalusan gambar biner.



Gambar 3.16. GUI axes6 dengan perintah penghalusan biner

Pada gambar 3.16. merupakan perintah untuk menampilkan proses penghalusan nilai tepi biner sehingga memudahkan dalam proses selanjutnya yaitu memisahkan background dengan objek yang dibutuhkan. Berikut ini merupakan source code yang digunakan pada proses ini.

```

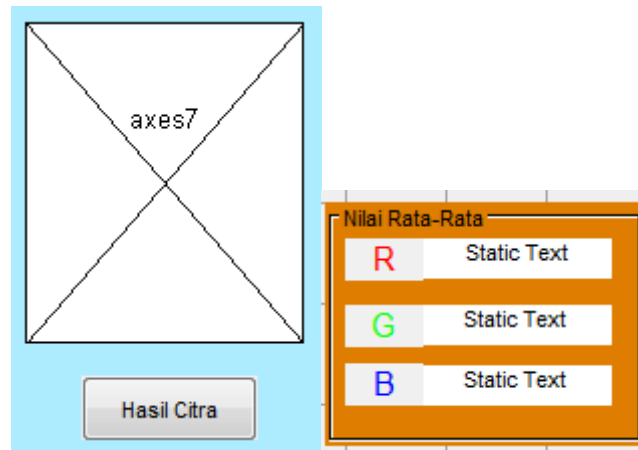
223 % --- Executes on button press in pushbutton6.
224 function pushbutton6_Callback(hObject, eventdata, handles)
225 % hObject handle to pushbutton6 (see GCBO)
226 % eventdata reserved - to be defined in a future version of MATLAB
227 % handles structure with handles and user data (see GUIDATA)
228
229 proyek=guidata(gcbo);
230 Ibw=get(proyek.axes5,'Userdata');
231 Ibw = imfill(Ibw,'holes'); %perbaikan garis tepi citra
232 Ibw = bwareaopen(Ibw,100); %untuk menghilangkan object kecil pada matriks
233 %citra dengan menghilangkan luasan yang kurang dari 100
234 str = strel('disk',5);
235 Ibw = imerode(Ibw,str);
236 set(proyek.figure1,'CurrentAxes',proyek.axes6);
237 set(imshow(Ibw));
238 set(proyek.figure1,'Userdata',Ibw);
239 set(proyek.axes6,'Userdata',Ibw);
240

```

Gambar 3.17. Source code untuk perintah penghalusan biner

Pada gambar 3.17. merupakan source code yang digunakan pada proses penghalusan citra biner fungsi tersebut digunakan untuk memperbaiki garis tepi citra dengan source code yang digunakan pada line 231 atau pada Matlab digunakan fungsi `imfill` dan pada line 232 terdapat fungsi untuk menghilangkan objek kecil citra dengan menghilangkan luasan citra yang nilainya kurang dari 100. Sehingga citra yang akan ditampilkan akan lebih halus pada tepi gambarnya yang memisahkan antara objek dan background dan tidak adanya objek kecil berwarna hitam pada bagian-bagian citra objek yang berwarna putih. Selanjutnya digunakan fungsi `strel` yang berguna sebagai elemen struktur atau penentu bentuk pada line 234 fungsi `strel` yang digunakan yaitu 'disk', 5 yang artinya elemen struktur berbentuk cakram dengan radius 5. Kemudian pada fungsi

imerode(Ibw,str) yaitu sebagai fungsi yang digunakan untuk operasi erosi dalam menyusutkan elemen dengan menggunakan elemen strel yang yang sebelumnya telah dibentuk atau strel('disk', 5) pada gambar Ibw. Berikut ini merupakan GUI hasil dari pengolahan citra yang telah dilakukan pada beberapa proses sebelumnya.



Gambar 3.18. GUI hasil citra dan hasil pembacaan nilai RGB.

Pada gambar 3.18. merupakan GUI yang akan menampilkan hasil dari keseluruhan proses pengolahan citra digital yang telah dilakukan pada proses-proses sebelumnya pada axes 7. Berikut ini merupakan source code yang digunakan pada push button hasil citra.

```

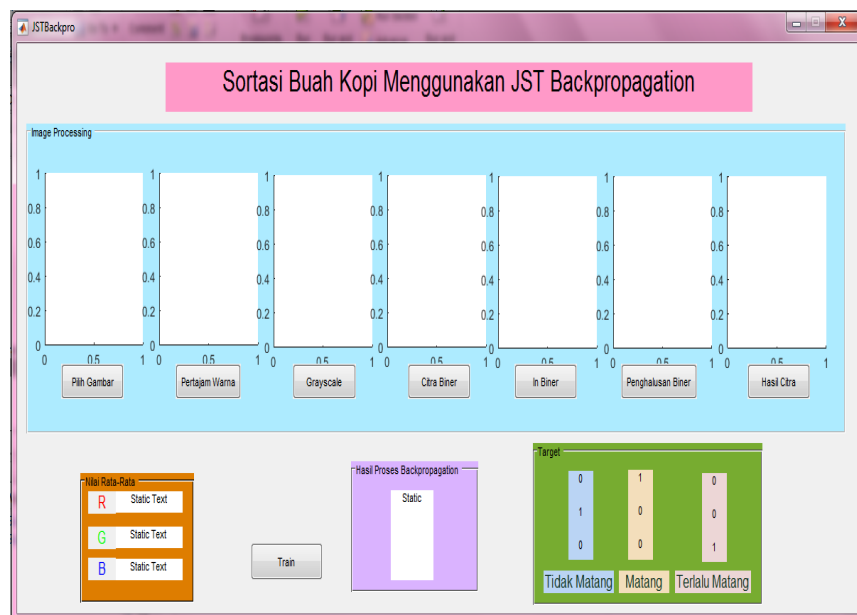
242 % --- Executes on button press in pushbutton7.
243 function pushbutton7_Callback(hObject, eventdata, handles)
244 % hObject    handle to pushbutton7 (see GCBO)
245 % eventdata  reserved - to be defined in a future version of MATLAB
246 % handles    structure with handles and user data (see GUIDATA)
247
248 proyek=guidata(gcbo);
249 edgeSharp=get(proyek.axes2,'Userdata');
250 Ibw=get(proyek.axes6,'Userdata');
251
252 R = edgeSharp(:,:,1);
253 G = edgeSharp(:,:,2);
254 B = edgeSharp(:,:,3);
255
256 R(~Ibw) =0; %pembacaan nilai tepi RGB
257 G(~Ibw) =0;
258 B(~Ibw) =0;
259
260 RGB_stretch = cat(3,R,G,B);
261
262 set(proyek.figure1,'CurrentAxes',proyek.axes7);
263 set(imshow(RGB_stretch));
264 set(proyek.figure1,'Userdata',RGB_stretch);
265 set(proyek.axes7,'Userdata',RGB_stretch);
266
267 sumI = sum(sum(RGB_stretch));
268 s = size(RGB_stretch);
269 rata_rataI = sumI./(s(1)*s(2));
270
271 rata_rataR = rata_rataI(1);
272 handles.rata_rataI(1)=rata_rataR;
273 rata_rataG = rata_rataI(2);
274 handles.rata_rataI(2)=rata_rataG;
275 rata_rataB = rata_rataI(3);
276 handles.rata_rataI(3)=rata_rataB;
277
278 guidata(hObject,handles);
279 set(handles.text3,'string',rata_rataR);
280 set(handles.text4,'string',rata_rataG);
281 set(handles.text5,'string',rata_rataB);
282

```

Gambar 3.19. Source code untuk perintah hasil

Pada gambar 3.19. source code yang digunakan untuk menampilkan hasil citra dengan meningkatkan kontras warna gambar dan menghilangkan background pada gambar. Fungsi yang digunakan pada source code diatas yaitu program memanggil 2 buah data yang disimpan yaitu data edgeSharp yang ditampilkan pada axes2 dan data Ibw yang ditampilkan pada axes6. Selanjutnya dilakukan pemisahan nilai Red, Green, dan Blue pada data edgeSharp seperti pada line 252-254, kemudian dilakukan pembacaan nilai tepi pada RGB menggunakan file Ibw dan yang terakhir pada line 260

digunakan perintah `cat(3,R,G,B)` yaitu menggabungkan kedua buah file sebelumnya yaitu `edgeSharp` dan `Ibw` menjadi satu gambar yang utuh dan ditampilkan pada `axes7`. Setelah itu dilakukan pembacaan nilai objek gambar berupa ilai Red, Green, dan Blue dengan rumus yang terdapat pada line 267-269. Fungsi rumus tersebut digunakan sebagai acuan untuk menentukan nilai rata-rata dari setiap warna yaitu Red, Green, dan Blue yang kemudian ditampilkan pada table yang telah disediakan pada tampilan GUI sistem ini. Berikut ini merupakan tampilan sistem pada saat dijalankan.



Gambar 3.20. Tampilan program saat dijalankan.

Pada gambar 3.20. merupakan tampilan sistem sortasi buah kopi pada saat dijalankan dalam Matlab. Pada tombol paling kiri merupakan perintah untuk menjalankan *pushbutton1* atau “Pilih Gambar” maka sistem dapat memilih ataupun mengambil gambar dengan format .bmp atau .jpeg dan file akan menyimpan nilai variabel kemudian ditampilkan pada `axes1`. Selanjutnya tombol kedua dari kiri yaitu fungsi *pushbutton2* atau “Pertajam warna”. Kemudian tombol *pushbutton3* atau “Grayscale” untuk mengubah



warna citra menjadi *greyscale*, fungsi *pushbutton4* atau “Citra Biner” untuk mengubah citra *grayscale* menjadi citra biner, fungsi *pushbutton5* atau “In Biner” untuk membalikkan citra objek dan citra *background*. Kemudian fungsi *pussbutton6* atau “Penghalusan Biner” untuk menghaluskan nilai tepi pada citra *in biner*, fungsi *pushbutton6* atau “Hasil Citra” untuk menampilkan citra yang telah dipertajam dan dihilangkan *backgroundnya*, serta membaca nilai warna citra yaitu *red*, *green* dan *blue*.

## **BAB IV**

### **HASIL DAN PEMBAHASAN**

#### **4.1. Pengujian Sistem**

##### **4.1.1. Proses Pengolahan Data Awal**

Hasil gambar yang diambil merupakan data sample yang akan digunakan pada tahap awal dalam *data preprocessing* pada proses ini data dilakukan pengolahan citra gambar dengan melakukan *scalling*, penajaman citra warna dengan metode CLAHE, kemudian dilakukan perubahan warna dengan menjadikan *grayscale*, dan tahap terakhir pada *preprocessing* ini yaitu mengubah gambar menjadi citra biner atau *black and white* dengan *thresholding*. Fungsi *preprocessing* ini dilakukan untuk mengolah citra gambar agar terpisah dari *backgroundnya* dan memiliki ketajaman warna *Red*, *Green*, dan *Blue* menjadi lebih terang pada setiap warnanya. Sehingga diharapkan sistem mampu membaca nilai setiap warna dengan maksima dan memperoleh data input RGB dengan baik.

#### 4.1.2. Pembagian Data

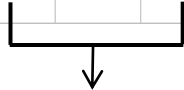
Pada penelitian ini mula-mula dilakukan penelitian langsung ke kebun kopi sinarwaya kemudian dilakukan pemetikan manual buah kopi. Setelah dilakukan pemetikan buah kopi dimasukkan di dalam sebuah wadah yang kemudian di ambil gambarnya. Selanjutnya dilakukan pembagian data sample dengan menetapkan target output, pembagian data sample pada penelitian ini terdiri dari dua buah data sample yaitu data pelatihan dan data uji. Pada sample data pelatihan terdapat 20 gambar sample yang terdiri dari 3 buah target output yaitu tidak matang, matang dan terlalu matang.

**Tabel 4.1.** Data Pelatihan sebagai input JST *backpropagation*

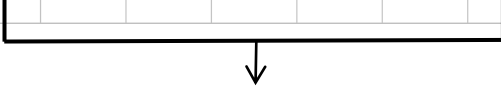
datalatih										
3x20 double										
	1	2	3	4	5	6	7	8	9	10
1	58.7292	79.9079	69.7420	49.3000	100.6770	97.4732	106.5370	113.7070	106.8420	107.9420
2	33.6973	47.1690	42.0879	52.8952	99.4531	96.8217	94.7425	99.4113	94.1731	100.1010
3	30.3250	41.0318	36.3620	26.3073	88.2878	82.0328	97.0862	101.5910	90.0138	98.1053
4										
	11	12	13	14	15	16	17	18	19	20
1	77.3568	61.1192	74.0657	111.8550	110.4250	56.1600	103.3440	60.8573	67.3848	105.9250
2	57.9125	43.1662	57.1615	95.6842	94.3713	37.3253	87.3161	50.3776	43.0869	91.6563
3	45.5931	31.6285	42.9505	101.1320	94.1496	31.0353	91.3788	28.8881	37.6929	96.1793
4										

**Tabel 4.2.** Target data pelatihan

datalatih target										
3x20 double										
	1	2	3	4	5	6	7	8	9	10
1	1	1	1	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1	1	1	1
3	0	0	0	0	0	0	0	0	0	0
4										

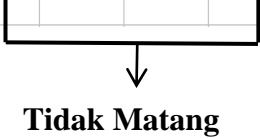


**Matang**

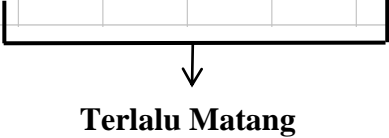


**Tidak Matang**

	11	12	13	14	15	16	17	18	19	20
1	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	0	0	0	0
3	0	0	0	0	0	0	1	1	1	1
4										



**Tidak Matang**



**Terlalu Matang**

Pada tabel 4.1. merupakan data pelatihan yang terdiri dari 20 data sample dengan masing-masing sample memiliki 3 buah nilai input yaitu nilai warna *red*, *green* dan *blue*. Masing-masing data tersebut didapat dari sample yang telah dilakukan *preprocessing* sehingga diharapkan data yang didapat cukup akurat dalam tingkat pengambilan warna pada objek yang diharapkan. Pada table 4.2. merupakan target keluaran output yang diharapkan pada data latih yang telah melakukan *preprocessing*. Data tersebut terdiri dari 3 buah target data bernilai matang atau output biner yang diharapkan bernilai 1 0 0 dengan kriteria penentuan gambar yang terdiri dari buah kopi yang berwarna merah terang secara manual, kemudian 12 buah output yang diharapkan bernilai tidak matang dengan nilai biner 0 1 0 dengan kriteria manual yaitu terdiri dari campuran warna buah kopi yaitu hijau, kekuninga dan merah, dan yang terakhir 5 buah target yang diharapkan bernilai terlalu matang dengan nilai biner 0 0 1 dan kriteria penentuan secara manual yaitu gambar yang terdiri dari buah kopi merah terang dan merah gelap.

#### 4.1.3. *Preprocessing*

Pada tahap ini mula-mula sistem akan di jalankan dan menampilkan GUI seperti yang telah dirancang. Kemudian dilakukan pengolahan citra digital (*preprocessing*) untuk meningkatkan warna dan menghilangkan background pada citra gambar. Pada *preprocessing* ini dilakukan 7 buah tahapan yang dilakukan.



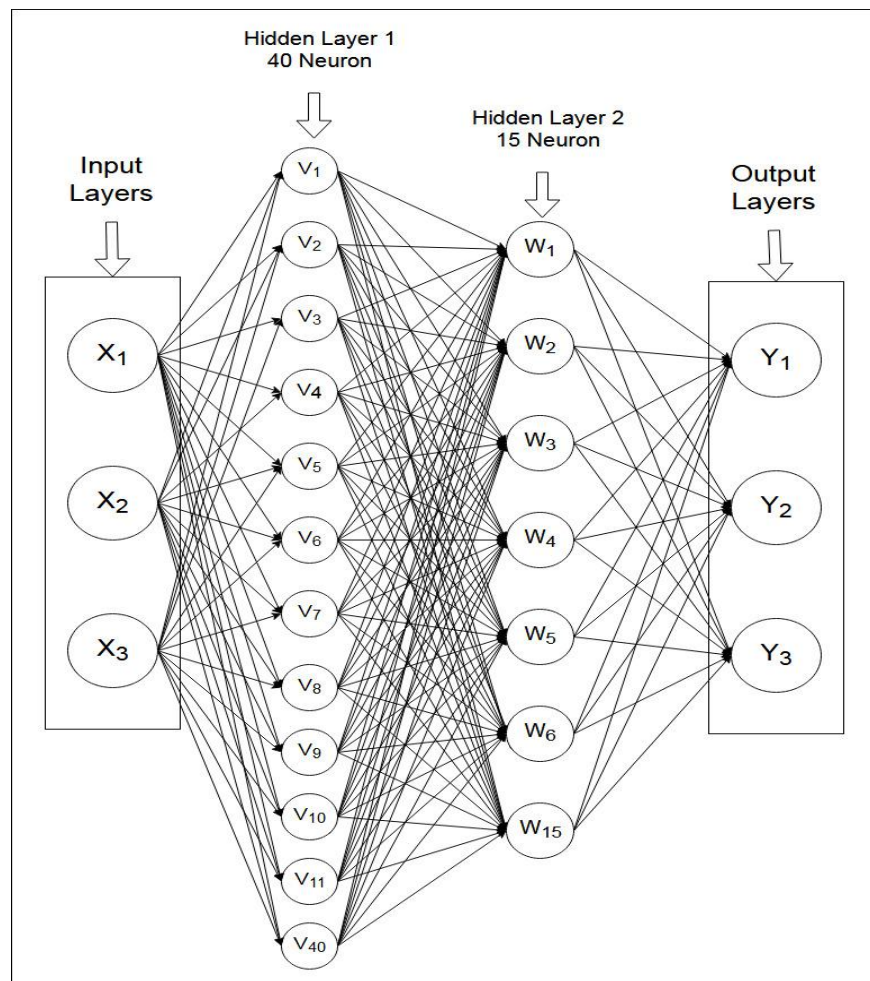
Gambar 4.1. Proses *preprocessing* sample pelatihan

Pada gambar diatas merupakan cara kerja sistem dengan tahapan awal yaitu *preprocessing* yaitu pengolahan citra gambar untuk mengambil nilai RGB yang akan digunakan sebagai input pada proses JST *backpropagation*. Dapat dilihat pada gambar 4.1. bahwa mula-mula kita akan diminta untuk memilih gambar yang akan diolah, kemudian dilakukan pertajaman warna, kemudian dilakukan proses pemisahan objek gambar dan *background* (*greyscale*, *bine*, *in bine* dan penghalusan *biner*). Terakhir dimunculkan citra gambar setelah dilakukan beberapa proses tersebut pada hasil citra dan membaca nilai dari warna gambar yaitu *red*, *green*, and *blue* seperti pada gambar 4.1.

#### 4.1.4. Model Arsitektur Jaringan

Arsitektur jaringan yang dibangun pada penelitian ini yaitu gambaran himpunan unit-unit yang dihubungkan dengan setiap jalur memiliki bobot yang berbeda beda . *Backpropagation* merupakan

salah satu metode yang terdapat pada jaringan syaraf tiruan, dengan kemampuan jaringan untuk mendapatkan keseimbangan antara mengenali pola yang digunakan selama pelatihan serta kemampuan jaringan untuk memberikan respon yang benar terhadap pola masukan serupa (tapi tidak sama) dengan pola yang dipakai selama pelatihan. Berikut ini merupakan arsitektur JST *backpropagation* yang akan dibangun.



Gambar 4.2. arsitektur jaringan yang dibangun.

Pada gambar diatas merupakan arsitektur jaringan *backpropagation* yang dibangun dengan 3 buah input layer, 2 hidden layer dengan tiap hidden layer berisi 40 neuron untuk hidden layer ke-1 dan 15 neuron untuk hidden layer ke-2, dan 3 buah output yang bernilai 0/1.

#### 4.1.5. Pelatihan

Pelatihan yang dilakukan pada penelitian ini yaitu menggunakan perangkat lunak (*software*) MATLAB yang telah menyediakan fungsi-fungsi pelatihan JST dengan algoritma *backpropagation*. Pada tahap pelatihan ini menggunakan jaringan *multi layer* (layar jamak) dengan 2 buah *hidden layer* dengan momentum. Pada MATLAB fungsi yang digunakan untuk membentuk jaringan adalah *newff* dengan format berikut ini :

$$net = newff(PR, [S1\ S2..\ SN], \{TF1\ TF2..\ FTN\}, BTF, BLF, PF)$$

dengan :

- $net$  = Jaringan *backpropagation* yang terdiri dari  $n$  layar.
- $PR$  = Matriks ordo  $R \times 2$  yang berisi nilai maksimum dan minimum.
- $S_i$  ( $i=1,..$ ) = Jumlah unit pada layar ke- $i$ .
- $T_i$  ( $i=1,..$ ) = Fungsi aktivasi pada layar ke- $i$ . Defaultnya (*sigmoid biner*).
- $BTF$  = Fungsi pelatihan jaringan . Defaultnya *traingdf*.
- $BLF$  = Fungsi perubahan bobot/bias. Defaultnya *learnrdf*.
- $PF$  = Fungsi perhitungan *performance/error*. Defaultnya *MSE*.

Berikut ini merupakan *syntax* yang digunakan pada penelitian ini menggunakan MATLAB.

```

76 % --- Executes on button press in pushbutton8.
77 function pushbutton8_Callback(hObject, eventdata, handles)
78 % hObject    handle to pushbutton8 (see GCBO)
79 % eventdata  reserved - to be defined in a future version of MATLAB
80 % handles     structure with handles and user data (see GUIDATA)
81
82 a = load('datalatih.mat'); %data training
83 t = load('target.mat'); %target
84 d = handles.rata_rataI(1);
85 e = handles.rata_rataI(2);
86 f = handles.rata_rataI(3);
87 c = [d;e;f];
88
89 %membangun jaringan
90 net=newff(minmax(a.datalatih),[40 15 3], {'logsig' 'logsig' 'logsig'},'trainrp', 'learnqdm', 'mse') %neural dan layer
91
92 %melihat bobot
93 bobotawal_input = net.IW(1,1) %bobot akan direndom oleh matlab
94 biasawal_input = net.b(1)
95 bobotunit_tersembunyi = net.LW(2,1)
96 biasunit_tersembunyi_input = net.b(2)
97 bobotunit_tersembunyi_ke_output = net.LW(3,2)
98 biasunit_tersembunyi_ke_input = net.b(3)
99
100
101 %penentuan parameter training
102 net.trainParam.epochs=1000 %jumlah iterasi
103
104 net.trainParam.lr=0.01 %learning rate
105 net.trainParam.mc=0.9 %momentum
106
107 %melakukan training
108 net=train(net,a.datalatih,t.target)
109

```

Gambar 4.3. Syntax program pelatihan *backpro* pada MATLAB

Pada gambar diatas digunakan 2 buah data dengan format .mat yang merupakan data pelatihan dan target seperti pada tabel yang telah dijelaskan pada sub bab sebelumnya. Selanjutnya pada baris line ke-84 sampai ke-86 merupakan fungsi pemanggilan data sebelumnya yang diambil pada program yaitu data nilai gambar yang diolah pada program yang sedang berjalan. Kemudian dilakukan perintah selanjutnya yaitu membangun jaringan dengan perintah JST yang terdapat pada MATLAB yaitu *newff* yang membuat model jaringan syaraf sesuai dengan matriks yang diharapkan pada *datalatih.mat* yaitu 3 buah input data, jumlah neuron *hidden layer* 1 berjumlah 40, jumlah neuron pada *hidden layer* 2 berjumlah 15, dan jumlah neuron pada output layer berjumlah 3, dengan menggunakan fungsi aktivasi *sigmoid biner (logsig)*. Dalam mempercepat pelatihan JST *backpropagation* digunakan fungsi *trainrp* (resilient



backpropagation) yang merupakan fungsi terbaik dalam penelitian ini yaitu dengan menggunakan fungsi aktivasi sigmoid biner maka akan menerima masukan dengan range tak berhingga menjadi keluaran  $[0,1]$ . Permasalahan yang terjadi apabila semakin jauh titik dari  $X=0$  dan semakin kecil gradiennya, sehingga hal ini menimbulkan masalah pada saat menggunakan metode penurunan tercepat (iterasi didasarkan atas gradien). Gradien kecil menyebabkan perubahan bobot juga kecil, meskipun jauh dari titik optimal. Pada permasalahan ini *trainrp* dapat mengatasinya dengan cara membagi arah perubahan bobot menjadi 2 bagian yang berbeda dan ketika menggunakan penurunan tercepat yang diambil hanya arahnya saja. Besarnya perubahan bobot dilakukan dengan cara lain. Fungsi *learnsgdm* dalam menentukan perubahan bobot dan bias pada pelatihan dan fungsi *mse* digunakan untuk menghitung *error*.

Selanjutnya menggunakan fungsi yang terdapat pada MATLAB yaitu inisialisasi bobot, MATLAB akan memberikan nilai bobot dan bias awal dengan bilangan acak kecil dan nilai tersebut akan berubah setiap kali membentuk jaringan. Maka digunakan fungsi *net.IW{1,1}* yang berguna untuk memberikan nilai bobot awal pada jaringan input ke jaringan tersembunyi ke-1 dan fungsi *net.b{1}* berguna untuk memberikan nilai bias secara acak. Selanjutnya fungsi *net.LW{2,1}* dan *net.b{2}* berfungsi untuk memberikan nilai bobot dari jaringan tersembunyi satu ke jaringan tersembunyi dua kemudian diberikan bias secara acak, begitu pula perintah selanjutnya. Penentuan parameter pelatihan yang di set sebelum dilakukan pelatihan yaitu :

- ✓ *net.train.Param.epoch* = 1000 digunakan untuk penentuan jumlah *epoch* pelatihan . Satu *epoch* merupakan satu siklus dengan melibatkan seluruh pola data *training (training pattern)*. Pada penelitian ini digunakan 1000 *epoch*.

- ✓ `net.trainParam.goal = 0.0001` yaitu parameter yang digunakan untuk menampilkan batas nilai *mse* agar iterasi dapat dihentikan. Apabila *mse* lebih kecil dari nilai batas yang ditentukan maka iterasi akan berhenti atau jumlah *epoch* mencapai batas yang ditentukan maka iterasi akan berhenti juga. Pada penelitian ini di set 0,0001.
- ✓ `net.trainParam.lr = 0.01` yaitu digunakan sebagai penentuan laju pemahaman. Pada penelitian ini di set sama seperti *default* 0.01.
- ✓ `net.trainParam.mc = 0.9` yaitu untuk menentukan momentum. Pada penelitian ini di set sesuai *default* 0.9.

Terakhir melakukan *training* dengan fungsi yang digunakan pada MATLAB yaitu.

```
net = train(net,a.datalatih,t.target)
```

dengan fungsi tersebut data akan melakukan pelatihan menggunakan jaringan syaraf *backpropagation* dengan datalatih dan target sebagai input dan output.

#### 4.1.6. Pengujian

Pada tahap pengujian digunakan sample baru yang telah di bedakan dengan sample pelatihan. Yang terdiri dari 15 buah sample. Nilai data sample dan target manual untuk data uji terdapat pada lampiran 1. Dalam JST *backpropagation* pengujian dilakukan untuk melihat jaringan apakah sudah dapat membaca pola pada pelatihan yang telah dilakukan dan menghasilk output yang sesuai dengan pola yang diharapkan. Berikut ini *syntax* yang digunakan untuk pengujian sistem pada MATLAB.

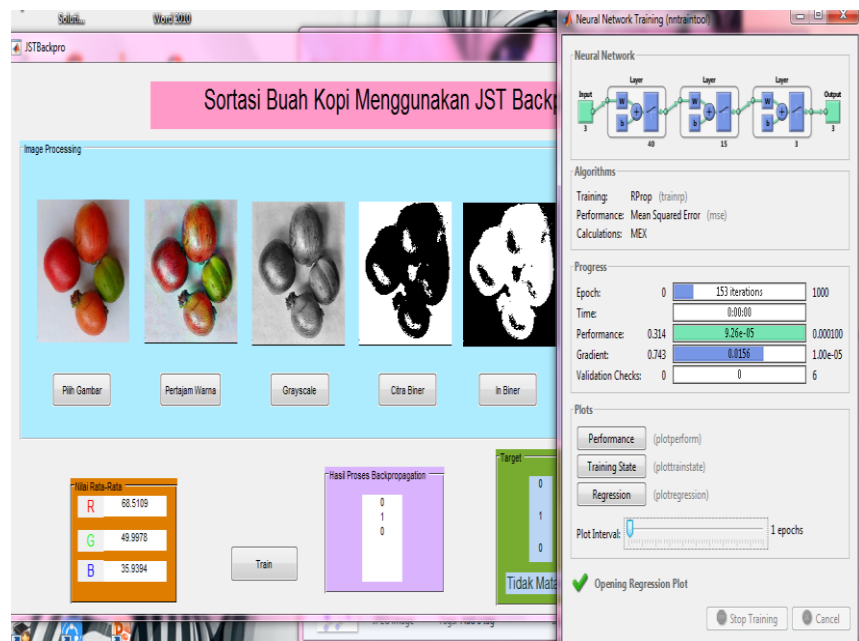
```

105 %melakukan testing
106 Y=sim(net,c)
107 Y=round(Y)
108 guidata(hObject,handles);
109 set(handles.text9,'string',Y);

```

Gambar 4.4. *yntax* program pengujian *backpro* pada MATLAB

Pada gambar 4.4. digunakan *syntax*  $Y = \text{sim}(\text{net}, c)$  dan  $Y = \text{round}(Y)$  maksudnya adalah bilangan  $Y$  = pengelompokan vektor jaringan  $c$  (data yang dijelaskan pada proses pelatihan) dan  $Y = \text{round}(Y)$  yang artinya pembulatan nilai  $Y$  ke bilangan bulat terdekat. Kemudian nilai akan ditampilkan pada `text9`. Berikut ini merupakan proses *testing* pada *JST backpropagation*.



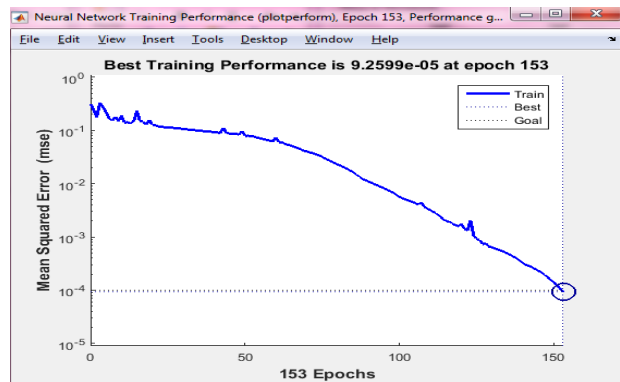
Gambar 4.5. Hasil pengujian *JST backpro*

Pada gambar 4.5. didapatkan hasil pengujian dengan rincian sebagai berikut :

1. Pada *epoch* pemberhentian iterasi ke-153 dari 1000 iterasi yang dijadikan parameter.
2. *Time* atau waktu yang dicapai pada iterasi ke-153 yaitu 0 detik.

3. *Error* yang dihasilkan tercapai dengan parameter yang diharapkan yaitu 0.000926.
4. Nilai gradiennya yaitu 0.0156.

Dari data yang telah didapat pada proses pengujian bahwasannya parameter-parameter yang diharapkan pada proses pelatihan telah tercapai dan berlangsung dengan cepat. Berikut ini merupakan grafik tingkat kesalahan (*error*) yang diperoleh pada proses pengujian ini.



Gambar 4.6. Grafik *train* pelatihan

Berdasarkan grafik diatas dapat dilihat bahwa grafik menunjukkan *performance* yang terbaik dengan tingkat *error* 0.000956 dan *epoch* pada iterasi ke-153 dengan waktu yang sangat cepat yaitu 0 detik. Pada grafik tersebut tidak adanya penurunan ataupun kenaikan yang terlalu tinggi ataupun terlalu rendah. Hal ini dikarenakan dilakunnya penambahan parameter momentum untuk mengurangi tingginya atau rendahnya penurunan yang terjadi pada saat *train*. Pada penelitian ini untuk nilai *epoc*, *time*, *error* dan gradien pada 15 sample pelatihan terdapat pada lampiran 1.

## **BAB V**

### **KESIMPULAN DAN SARAN**

#### **5.1. Kesimpulan**

Pada penelitian ini jaringan syaraf tiruan *backpropagation* yang digunakan sebagai sistem penyortiran buah kopi menunjukkan tingkat akurasi benar mencapai 86,66%. Dengan menggunakan 15 samaple uji yang digunakan sebagai data pelatihan. Hal tersebut dapat menunjukkan bahwa tingkat kesalahan yang terjadi pada sistem sortasi buah kopi ini cukup kecil yaitu 13,33% . Sehingga dapat disimpulkan bahwa penelitian untuk membuat sistem sortasi buah kopi menggunakan JST *backpropagation* dianggap sudah cukup baik dalam melakukan pelatihan dan pengenalan pola dalam melakukan sortasi buah kopi menggunakan Matlab sehingga pada penelitian ini sudah dapat mencapai visi teknik informatika unila dalam kekhasan pada bidang *ICT in Agriculture*. Adapun kelemahan yang terdapat pada penelitian ini dalam pembuat sistem sortasi buah kopi yaitu sitem yang di bangun dianggap kurang *real time* dalam pengolahan hasil output yang dihasilkan. Kemudian apabila dilakukan pengujian lebih lanjut pada data pengujian terdapat beberapa data uji yang mengalami perubahan nilai output yang terkadang tidak sesuai dengan output yang diharapkan ataupun sangat berbeda. Berdasarkan hasil penelitian ini diharapkan memudahkan para petani kopi dalam proses sortasi buah kopi dan dapat dikembangkan oleh pihak-pihak terkait dalam pengembangannya sehingga dapat meningkatkan dan menghasilkan mutu kopi terbaik.

## 5.2. Saran

Berikut ini merupakan beberapa saran untuk menyempurnakan penelitian ini yaitu :

1. Penambahan data pelatihan dan data uji untuk buah kopi yang matang sempurna dan terlalu matang.
2. Perlunya peningkatan penggunaan sistem dengan cara pengambilan citra gambar secara *real time* sehingga memudahkan pengguna pada saat proses sortasi karena buah kopi yang telah dipetik tidak dapat terlalu lama dibiarkan karena dapat mengurangi mutu buah kopi.

## DAFTAR PUSTAKA

- [1] Direktorat Jendral Perkebunan. 2015. *Statik Perkebunan Indonesia*.
- [2] Kementerian Pertanian Direktorat Jenderal Perkebunan. 2014. *Pedoman Teknis Budidaya Kopi yang Baik (Good Agriculture Practices/GAP On Coffee)*.
- [3] Noor, M. Helmi dan Moch. Hariadi. 2009. *Image Cluster Berdasarkan Warna Untuk Identifikasi Kematangan Buah Tomat Dengan Metode Valley Tracing*. Surabaya: ITS.
- [4] Kastaman, Roni dan Fadhil Abdulfatah. 2009. *Analisis Kinerja Perangkat Lunak Pengolahan Citra dengan Menggunakan Beberapa Metode Klasifikasi untuk Menentukan Kualitas Buah Manggis*. Bandung : Universitas Padjajaran.
- [5] Jumarwanto, A. Hartanto, R. Prastiyanto, D. 2009. *Aplikasi Jaringan Syaraf Tiruan Backpropagation untuk Memprediksi Penyakit THT di Rumah Sakit Mardi Rahayu Kudus*. [www.elektroindonesia.com/jst](http://www.elektroindonesia.com/jst) diupload tanggal 28 Agustus 2007.
- [6] Siang, Jong J. 2005. *Jaringan Syaraf Tiruan dan Pemrogramannya menggunakan Matlab*. Andi. Yogyakarta.
- [7] Cronquist, A. 1981. *An Integrated System Of Classification Of Flowering Plants*. New York : Columbia University Press.
- [8] Wachjar, A. 1984. *Pengantar Budidaya Kopi*. Bogor : Fakultas pertanian.
- [9] Najiyati, S dan Danarti. 2001. *Kopi Budidaya dan Penanganan Lepas Panen*. Jakarta :: Penebar Swadaya Jakarta.
- [10] Rahardjo, Pudji. 2012. *Panduan Budidaya dan Pengolahan Kopi Arabika dan Robusta*. Jakarta. Penebar Swadaya.

- [11] Ciptadi dan MZ Nasution. 1985. *Pengolahan Kopi*. Bogor. Agro Industri Press.
- [12] Putra, Darma. 2010. *Pengolahan Citra Digital*. Andi Offset. Yogyakarta.
- [13] Microsoft Windows. *Microsoft Paint*. 10 Oktober 2018.  
[https://en.wikipedia.org/wiki/Microsoft\\_Paint](https://en.wikipedia.org/wiki/Microsoft_Paint)
- [14] Madenda, Sarifuddin. 2015. *Pengolahan Citra dan Video Digital*. Erlangga. Jakarta.
- [15] Kusumanto, RD dan Tompunu, Alan Novi. 2011. *Pengolahan Citra Digital Untuk Mendeteksi Obyek Menggunakan Warna Model Normalisasi RGB*. Politeknik Negeri Sriwijaya. Palembang.
- [16] Deswari, Dila. Hendric. Derisma. 2010. *Identifikasi Komatangan Buah Tomat Menggunakan Metode Backpropagation*. Politeknik Negeri Padang. Padang.
- [17] Arham, Zaenul dkk. 2004. *Evaluasi Mutu Jeruk Nipis (Citrus Aurantifolia Swingle) Dengan Pengolahan Citra Digital Dan Jaringan Syaraf Tiruan*. Bogor. IPB.
- [16] Rapidtables. *RGB Color*. 10 Oktober 2018.  
[https://www.rapidtables.com/web/color/RGB\\_Color.html](https://www.rapidtables.com/web/color/RGB_Color.html).
- [17] Tim Parker, Rhodes.Fay,dkk. 2002. *Flash MX Designer's Action Script Reference*. Friens of ED.
- [18] Irwan, Muhamad. Sianipar, RH. 2018. *Pengantar Pengolahan Citra Digital*. Sparta Publisher.
- [19] MathWorks.*Adaptive Histogram Equalization*. 11 November 2018.  
<https://www.mathworks.com/help/images/adaptive-histogram-equalization.html>.
- [20] Nugraha, Drs Suwandi M. Si, Bethaningtyas. 2015. *Sistem Otomasi Dalam Penyortiran Tomat Dengan Image Processing Menggunakan Metode Deteksi RGB*. Bandung. Universitas Telkom.



- [21] Harjoko, Agus. 2014. *Pemrosesan Citra Digital Untuk Klasifikasi Mutu Buah Pisang Menggunakan Jaringan Saraf Tiruan*. Yogyakarta. Gadjah Mada University.
- [22] Indriana, Dina dan Nurtatio A, Pulung. 2015. *Peningkatan Kontras Menggunakan Metode Contrast Limited Adaptive Histogram Equalization Pada Citra Underwater*. Jawa Tengah. Universitas Dian Nuswantoro.

## LAMPIRAN 1

Tabel 1.data uji JST *backpropagation*

Data Uji Ke-	Input			JST <i>Backpropagation</i>				Output			Keterangan
	R	G	B	<i>Epoch</i>	<i>Time</i>	<i>Perfomance</i>	Gradien	Y1	Y2	Y3	
1.	71,3579	47,1765	41,5884	139 <i>Iteration</i>	0 s	0,000097	0,00585	0	0	1	Salah
2.	107,383	91,7108	94,5701	137 <i>Iteration</i>	0 s	0,000099	0,0115	0	0	1	Salah
3.	68,5109	49,9978	35,9394	115 <i>Iteration</i>	0 s	0,000086	0,0111	0	1	0	Benar
4.	44,2619	51,6776	16,4112	108 <i>Iteration</i>	0 s	0,000096	0,0094	0	1	0	Benar
5.	101,324	98,8908	86,1351	112 <i>Iteration</i>	0 s	0,000088	0,0067	0	1	0	Benar
6.	113,204	104,401	101,101	113 <i>Iteration</i>	0 s	0,000094	0,0315	0	1	0	Benar
7.	76,9535	54,1444	40,5937	130 <i>Iteration</i>	0 s	0,000082	0,0098	0	1	0	Benar
8.	67,9765	47,7412	35,591	113 <i>Iteration</i>	0 s	0,000083	0,0123	0	1	0	Benar
9.	67,2971	56,685	34,934	123 <i>Iteration</i>	0 s	0,000084	0,0090	0	1	0	Benar
10.	108,316	95,477	97,114	126 <i>Iteration</i>	0 s	0,000085	0,0114	0	1	0	Benar
11.	104,793	94,9004	95,5172	108 <i>Iteration</i>	0 s	0,000098	0,0045	0	1	0	Benar

12.	111,401	96,5014	95,8366	115 <i>Iteration</i>	0 s	0,000087	0,0210	0	1	0	Benar
13.	76,4663	51,2432	39,8329	120 <i>Iteration</i>	0 s	0,000076	0,0049	0	1	0	Benar
14.	72,66	47,5369	47,9851	127 <i>Iteration</i>	0 s	0,000087	0,0083	0	0	1	Benar
15.	108,443	89,9567	96,0358	153 <i>Iteration</i>	0 s	0,000092	0,0156	0	0	1	Benar