JTRISTE, Vol.2, No.1, Maret 2015, pp. 1~12
ISSN: 2355-3677

Implementasi Jaringan Syaraf Tiruan dengan Algoritma Propagasi Balik untuk Pengenalan Pola Angka

Abdul Tahir
Akademi Teknik Soroako
E-mail: abdultahir@ats-sorowako.ac.id

Abstrak

Pengenalan angka identik dengan beberapa penelitian yang berkaitan dengan pengenalan pola, terutama yang menggunakan teknik klasifikasi Jaringan Syaraf Tiruan (JST). Penelitian ini bertujuan untuk merancang aplikasi pengenalan angka dengan menerapkan algoritma JST Propagasi Balik. Untuk mendapatkan pola pengenalan yang terbaik, dilakukan sejumlah tahap. Pertama, pengumpulan data dengan mengambil ciri dari karakter angka. Kedua, dilakukan segmentasi pada masing-masing karakter dengan ukuran 10 x 8 pixels, setiap citra hasil segmentasi selanjutnya diubah kedalam format citra hitam putih dan direpresentasikan dalam bentuk biner 0-1, nilai biner dari setiap karakter dibentuk menjadi sebuah vektor sebagai input pada proses JST. Ketiga, menyusun arsitektur JST untuk proses pelatihan dan pengujian. Jumlah data yang digunakan dalam penelitian ini sebanyak 1400 data, 70% digunakan sebagai data training dan 30% digunakan sebagai data testing. Dari hasil ini diperolah model pengenalan JST terbaik dengan menggunakan 80 lapisan masukan, 30 lapisan tersembunyi dan 10 lapisan keluaran, model ini mampu menghasilkan akurasi validasi 99,49% dan pengujian 97,62%.

Kata kunci: Propagasi,Syaraf, Biner, citra, klasifikasi

Abstract

The number recognition identical with some research related to pattern recognition, especially the use of classification techniques Artificial Neural Network (ANN). This research aims to design application of number recognition using Artificial Neural Network (ANN) with back propagation algorithm. To get the best pattern recognition, be done some of stages. First, collected of data by taking the characteristics of numeric characters. Second, done segmentation on each character with a size of 10 x 8 pixels, each image segmentation then converted into a black and white image format and represented in binary form 0-1, the binary value of each character formed into a vector as an input to the process of the ANN . Third, prepared ANN architecture for training and testing process. The amount of data used in this research were 1400 data, the 75% is used as training data and 30% is used as a data testing. The results obtained from the introduction of ANN models with up to 80 layers of inputs, 30 hidden layers and 11 output layers, the model was able to produce validation accuracy of 99.49% and 97.62% testing.

Kata kunci: Back Propagation, Neuron, Binary, image, calsification

1. Pendahuluan

Teknik identifikasi pola karakter berkembang dengan pesat, terutama yang menggunakan teknik klasifikasi Jaringan Syaraf Tiruan (JST). Berikut ini adalah beberapa penelitian yang membahas tentang pengenalan karakter menggunakan JST. (1)Penerapan *neural network* tentang metode *backpropagation* pada pengenalan pola huruf [5], penelitian ini mampu mengenali pola huruf abjad dengan baik meskipun diberi noise dengan batas-batas tertentu. (2)Teknik pengenalan huruf menggunakan model jaringan syaraf tiruan *radial basis function* dengan *randomize cluster decision* [4]. Penelitian ini mengembangkan perpaduan model jaringan saraf tiruan antara model radial dengan pembentukan *cluster* menggunakan model SOM (*Self Organizing Map*). Hasil yang diperoleh menunjukkan bahwa lebih dari 97% (atau hampir 100%) huruf dapat dikenali oleh sistim berbasis radial tersebut. (3)Teknik pengenalan karakter berbasis optik yaitu *Optical Character Recognition System Using BP Algorithm* [1], penelitian ini menggunakan algoritma JST propagasi balik dengan tiga buah lapisan (layer), penelitian ini berhasil melakukan otomatisasi pengenalan karater secara efisien. Selain teknik

JST, beberapa penelitian juga menggunakan teknik lain untuk pengenalan karakter huruf dan angka, seperti algoritma genetika [6],dan perpaduan algoritma berbasis *chain code* dan *sequence allignment* [7]. Dalam penelitian ini metode pengenalan angka mengambil teknik klasifikasi JST dengan menggunakan alogoritma propagasi balik seperti yang dilakukan oleh Park [1], teknik ini diimplementasikan dengan menggunakan aplikasi yang menggunakan Pemrograman *Visual Basic*. Adapun yang mendasari pemilihan JST karena teknik ini memiliki kemampuan melakukan proses pembelajaran untuk menciptakan suatu pola pengetahuan dan perhitungan secara paralel sehingga prosesnya lebih singkat.

2. Teori Dasar

2.1 Pengolahan Citra

Bidang pengolahan citra secara digital mulai diminati diawal tahun 1921, yaitu saat pertama kali sebuah foto berhasil ditransmisikan secara digital melalui kabel laut dari kota New York ke kota London [7]. Dengan perkembangan teknologi komputer yang sanggup memenuhi kecepatan proses berbagai algoritma pengolahan citra, saat ini bayak dikembangkan berbagai aplikasi pengolahan citra yang secara umum dapat dikelompokkan dalam dua bagian, (1) Memperbaiki kualitas suatu gambar (citra) sehingga dapat lebih mudah diinterpretasikan oleh manusia, (2) Mengolah informasi yang terdapat pada gambar (citra) untuk keperluan pengenalan objek secara otomatis oleh suatu mesin. Pada dasarnya pengolahan citra *digital* menunjuk pada pemrosesan gambar dua dimensi menggunakan computer. Citra digital merupakan sebuah larik (array) yang berisi nilai-nilai riil maupun komplek yang dapat direpresentasikan dengan deretan bit tertentu. Citra dapat didefenisikan dengan sebuah fungsi f(x,y) berukuran matrik M kali N, dimana M adalah baris dan N adalah kolom serta x dan y merupakan koordinat spasial (Putra 2010).

2.1.1 Jenis Citra.

Nilai suatu *pixel* memiliki nilai dalam rentang tertentu, dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbeda-beda tergantung dari jenis warnanya. Namun secara umum jangkauannya adalah 0 - 255. Citra dengan penggambaran seperti ini digolongkan kedalam citra *integer*. Dalam ilmu pengolahan citra digital dikenal beberapa jenis citra yaitu : citra *biner*, *grayscale*, citra warna 8 bit, citra warna 16 bit, dan citra warna 24 bit. Citra *biner* merupakan citra digital yang hanya memiliki dua kemungkinan nilai *pixel* yaitu hitam dan putih. Citra biner juga disebut sebagai citra B&W (*black and white*) atau citra monokrom [11]. Hanya dibutuhkan 1 bit untuk mewakili nilai setiap *pixel* dari citra biner. Citra biner seringkali muncul sebagai hasil dari proses pengolahan seperti segmentasi, pengambangan, morfologi, ataupun *dithering*. Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pixelnya, dengan kata lain nilai bagian warna *Red(R)* = *Green(G)* = *Blue(B*). Nilai tersebut digunakan untuk menunjukkan tingkat intensitas. Warna yang dimiliki adalah warna hitam, keabuan dan putih. Tingkat keabuan disini merupakan warnah abu dengan berbagai tingkatan dari hitam hingga mendekati putih. Citra *grayscale* memiliki kedalaman warna 8 bit (256 kombinasi warna keabuan).

2.1.2 Segmentasi Citra

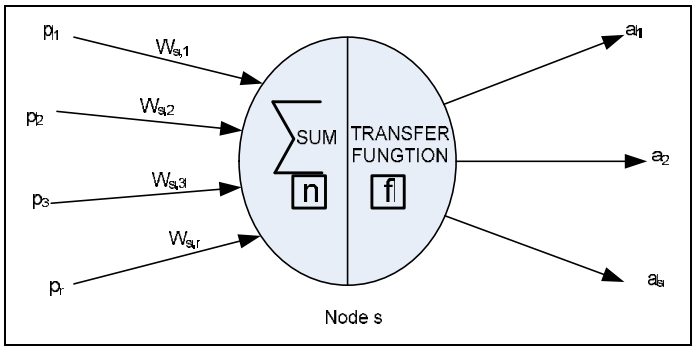
Segmentasi merupakan teknik untuk membagi suatu citra menjadi beberapa daerah (*region*) dimana setiap daerah memiliki kemiripan atribut. Pada penelitian ini teknik segmentasi yang digunakan adalah teknik pengembangan (*thresholding*). Teknik *theresholding* menghasilkan citra biner, citra yang memiliki dua nilai tingkat keabuan yaitu hitam dan putih atau dengan kata lain mengubah data pada citra agar hanya memiliki nilai 0 dan 1 [11]. Hal ini dilakukan untuk mempermudah mengetahui apakah pixel tersebut “terisi” atau tidak. Dengan menggunakan *thresholding* maka derajat keabuan bisa diubah sesuai keinginan.

2.2 Jaringan Syaraf Tiruan untuk Pengenalan Pola.

Jaringan Syaraf Tiruan (JST) atau *Artificial Neural Network* (ANN) adalah model jaringan neural yang meniru prinsip kerja dari *neuron* otak manusia (*neuron* biologis). JST pertama kali muncul setelah model sederhana dari *neuron* buatan diperkenalkan oleh *McCulloch* dan *Pitts* pada tahun 1940 [12]. Model sederhana tersebut dibuat berdasarkan fungsi *neuron* biologis yang merupakan dasar unit pensinyalan dari sistim syaraf. JST memiliki beberapa kemampuan seperti yang dimiliki oleh manusia, yaitu: Kemampuan untuk belajar dari pengalaman, Kemampuan melakukan perumpamaan (*generalization*) terhadap input baru dari pengalaman yang dimiliki, dan Kemampuan memisahkan (*abstraction*) karakteristik penting dari input yang mengandung data yang tidak penting.

2.2.1 Model JST

Prinsip kerja JST didasari pada mekanisme kerja penyaluran informasi sistim jaringan syaraf. Namum demikian karena keterbatasan yang dimiliki oleh struktur JST maka hanya sebagian kecil saja dari kemampuan sistim syaraf manusia yang dapat ditiru [12]. Gambar 2.2.1 berikut merupakan ilustrasi model JST.



Gambar 2.2.1 Ilustrasi model JST [12].

Berikut penjelasan Gambar 2.2.1:

- Pr menyatakan sinyal *input* dari node input ke i = 1,2,...,r, dengan r menyatakan jumlah input.
- $W_{s,r}$ menyatakan bobot (*weight*) hubungan dari node (neuron) *input* r ke node (neuron) yang di tuju j, j = 1,2,...,S, dengan S menyatakan jumlah neuron.
- n menyatakan total (jumlah) sinyal terbobot yang masuk ke node s atau juga sering disebut sebagai tingkat pengaktifan (*activation level*) di node s.
- f menyatakan fungsi transfer (*transfer fungtion*) yang menentukan keluaran dari node s dan tergantung dari nilai n.
- a_s menyatakan sinyal yang keluar (*outgoing signal*) atau *output* dari node s.

Nilai n dari model diatas dihitung dengan rumus :

$$n = W_{s,r} . p_r$$

sedangkan keluaran node yang dinyatakan dengan a dapat ditentukan sebagai berikut .

$$a = f(n)$$

seringkali kedua formula diatas digabung menjadi satu seperti berikut :

$$a = f(W_{s,r} p_r)$$

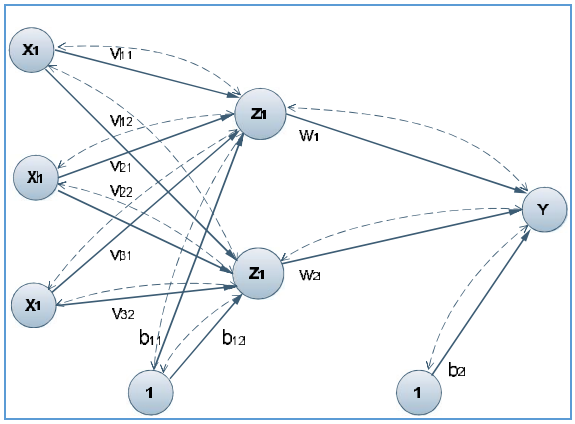
2.2.2 Pembelajaran pada JST

Proses pembelajaran (*learning*) atau pelatihan (*training*) pada JST merupakan proses perubahan atau penyesuaian tingkat kekuatan hubungan antar node-node yang saling terhubung [12]. Tingkat kekuatan hubungan antar node dinyatakan dengan nilai bobot. Ini berarti proses pembelajaran pada JST tidak lain merupakan proses penyesuaian nilai-nilai bobot tersebut. Proses pembelajaran merupakan suatu proses iterasi pada sistim JST yang cukup kompleks dan proses belajar membutuhkan waktu yang cukup panjang. Selama proses belajar faktor bobot mengalami perubahan dan bila tahapan belajar sudah selesai maka nilai-

nilai faktor bobot yang dihasilkan disimpan dan digunakan sebagai faktor bobot terpakai. Keandalan suatu JST tergantung pada keberhasilan dalam menemukan faktor bobot terpakai tersebut.

2.2.3 Algoritma JST Propagasi Balik.

Propagasi balik (*backpropagation*) adalah salah satu pengembangan dari arsitektur *Single Layer Neural Network*. Arsitektur ini terdiri dari *input layer*, *hidden layer* dan *output layer* [12]. Setiap layer terdiri dari satu atau lebih *artificial neuron*. Algoritma propagasi balik merupakan salah satu teknik pembelajaran terawasi (*supervised learning*) dan digunakan dalam eksperimen/penelitian ini, oleh karena itu diperlukan pemahaman beberapa unsur penting dalam metode propagasi balik. Di dalam jaringan propagasi balik, setiap unit yang berada di lapisan input terhubung dengan setiap unit yang ada di lapisan tersembunyi. Setiap unit yang ada di lapisan tersembunyi terhubung dengan setiap unit yang ada di lapisan *output*. Jaringan ini terdiri dari banyak lapisan (*multilayer network*). Ketika jaringan diberikan pola masukan sebagai pola pelatihan, maka pola tersebut menuju unit-unit lapisan tersembunyi untuk selanjutnya diteruskan pada unit-unit lapisan keluaran. Kemudian unit-unit lapisan keluaran memberikan respon sebagai keluaran JST. Saat hasil keluaran tidak sesuai dengan yang diharapkan, maka keluaran disebarkan mundur (*backward*) pada lapisan tersembunyi kemudian dari lapisan tersembunyi menuju lapisan masukan. Arsitektur jaringan propagasi balik seperti terlihat pada Gambar 1.2.2, x1 sampai dengan xn adalah *input layer*, z1 sampai dengan zp adalah *hidden layer* , dan y1 sampai dengan ym adalah output layer.



Gambar 2.2.2 Arsitektur jaringan propagasi balik [9].

Tahap pelatihan ini merupakan langkah untuk melatih suatu JST, yaitu dengan cara melakukan perubahan bobot. Sedangkan penyelesaian masalah dilakukan jika proses pelatihan tersebut telah selesai, fase ini disebut fase pengujian. Pada proses pelatihan algoritma propagasi balik menggunakan error output untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan error ini, tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu. Pelatihan propagasi umpan balik berbasis jaringan syaraf tiruan meliputi 3 fase. Fase pertama adalah fase maju. Pola masukan dihitung maju mulai dari lapisan masukan hingga lapisan keluaran menggunakan fungsi aktivasi yang ditentukan. Fase kedua adalah fase mundur. Selisih antara keluaran jaringan dengan target yang diinginkan merupakan kesalahan yang terjadi. Kesalahan tersebut dipropagasikan mundur, dimulai dari garis yang berhubungan langsung dengan unit-unit di lapisan keluaran. Fase ketiga adalah modifikasi bobot untuk menurunkan kesalahan yang terjadi. Berikut penjelasan umum setiap fase.

Fase I: Propagasi Maju Selama propagasi maju, sinyal masukan (xi) dipropagasikan ke lapisan tersembunyi menggunakan fungsi aktivasi yang ditentukan. Keluaran dari setiap unit lapisan tersembunyi (zj) tersebut selanjutnya dipropagasikan maju lagi ke layer tersembunyi di atasnya menggunakan fungsi aktivasi yang ditentukan. Demikian seterusnya hingga menghasilkan

keluaran jaringan (yk). Berikutnya, keluaran jaringan (yk) dibandingkan dengan target yang harus dicapai (tk). Selisih dari tk terhadap yk yaitu tk-yk adalah kesalahan yang terjadi. Jika kesalahan ini lebih kecil dari batas toleransi yang ditentukan, maka iterasi dihentikan. tetapi apabila kesalahan masih lebih besar dari batas toleransinya, maka bobot setiap garis dalam jaringan dimodifikasi untuk mengurangi kesalahan yang terjadi.

Fase II: Propagasi Mundur
Berdasarkan kesalahan tk-yk, dihitung faktor δk (k=1,2,..., m) yang dipakai untuk mendistribusikan kesalahan di unit yk ke semua unit tersembunyi yang terhubung langsung dengan yk. δk juga dipakai untuk mengubah bobot garis yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor δj (j = 1,2,..., m) di setiap unit di lapisan tersembunyi sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di lapisan di bawahnya. Demikian seterusnya hingga semua faktor di unit tersembunyi yang berhubungan langsung dengan unit masukan dihitung.

Fase III: Perubahan Bobot
Setelah semua faktor dihitung, bobot semua garis dimodifikasi bersamaan. Perubahan bobot suatu garis didasarkan atas faktor neuron di lapisan atasnya. Sebagai contoh, perubahan bobot garis yang menuju ke lapisan keluaran didasarkan atas k yang ada di unit keluaran. Ketiga fase terebut diulang-ulang terus hingga kondisi penghentian dipenuhi. Umumnya kondisi penghentian yang sering dipakai adalah jumlah iterasi atau kesalahan. Iterasi dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan.

2.3 Pengolahan Citra Digital Menggunakan *Visual Basic*

Visual Basic (atau sering disingkat VB) adalah perangkat lunak untuk menyusun program aplikasi yang bekerja dalam lingkup sistim operasi Windows. Dalam *Visual Basic* terdapat sebuah *toolbox* yang dapat digunakan untuk membuat objek sesuai kebutuhan. Objek penting yang banyak digunakan untuk keperluan pengolahan citra adalah *Picturebox*. Untuk menangkap citra yang ada pada *PictureBox* digunakan perintah *namaPicture*. *Point(x,y),* perintah ini menangkap warnah pada posisi (x,y), dan hasilnya adalah nilai warna dalam 224 warna RGB [10].

2.3.1 Mengubah Citra Berwarna Menjadi *Grayscale*

Proses awal yang banyak digunakan dalam pengolahan citra adalah mengubah citra berwarna menjadi citra grayscale, hal ini diperlukan untuk menyederhanakan model citra. Untuk mengubah citra berwarna menjadi citra *grayscale* dapat dilakukan dengan mengambil rata-rata nilai R, G, dan B [10]. Gambar dan penulisan code dalam *Visual Basic* dapat dilihat pada Gambar 2.3.1 berikut ini.

```
For I = 0 To Picture1.ScaleWidth Step 15
   For j = 0 To Picture1.ScaleHeight Step 15
        warna = Picture1.Point(I, j)
        r = warna And RGB(255, 0, 0)
        g = Int(warna And RGB(0, 255, 0)) / 256
        b = Int((Int(warna And RGB(0, 0, 255)) / 256) / 256)
        x = (r + g + b) / 3
        Picture2.PSet (I, j), RGB(x, x, x)
   Next j
```

Gambar 2.3.1 Konversi citra warna ke citra *grayscale* dalam *Visual Basic*

2.3.2 Konversi Citra *Grayscale* ke citra Biner

Citra biner (hitam-putih) merupakan citra yang banyak dimamfaatkan untuk keperluan pengenalan pola yang sederhana. Untuk mengubah suatu citra *grayscale* menjadi citra biner

dilakukan dengan mengubah kuantisasi citra atau proses *Thresholding*. Gambar dan penulisan *code* dalam *Visual Basic* dapat dilihat pada Gambar 2.3.2 berikut ini.

```
For I = 0 To Picture1.ScaleWidth Step 15
   For j = 0 To Picture1.ScaleHeight Step 15
        warna = Picture1.Point(I, j)
        r = warna And RGB(255, 0, 0)
        g = Int(warna And RGB(0, 255, 0)) / 256
        b = Int((Int(warna And RGB(0, 0, 255)) / 256) / 256)
        x = (r + g + b) / 3
        If x < 128 Then x = 0 Else x = 255
        Picture2.PSet (I, j), RGB(x, x, x)
    Next j
Next I
```

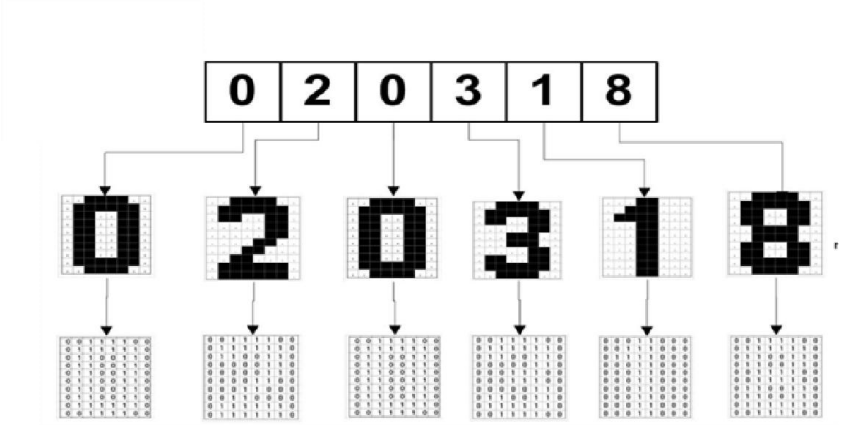
Gambar 2.3.2 Konversi citra *grayscale* ke citra biner dalam *Visual Basic*

3. Metodologi Penelitian

Metodologi penelitian dibagi dalam tiga tahapan yaitu persiapan penelitian dengan pengumpulan data, disain pemodelan JST untuk pengenalan angka, dan Implementasi JST dengan rancangan Aplikasi.

3.1 Persiapan Penelitian

Persiapan penelitian dilakukan mulai dengan pengumpulan data dengan mengambil ciri dari karakter yang dijadikan bahan dalam penelitian ini yaitu Angka 0 s/d 9. Dalam penelitian ini data dibagi dua, 70% digunakan untuk data *training* dan 30% digunakan untuk data pengujian. Langkah pertama yang dilakukan adalah melakukan segmentasi pada masing-masing karakter dengan ukuran 10 x 8 pixels. Setiap citra hasil segmentasi selanjutnya diubah kedalam format citra hitam putih. Citra setiap karakter direpresentasikan dalam bentuk *biner*, yang berwarna hitam diberi nilai 1 dan warna putih diberi nilai 0. Nilai *biner* dari setiap karakter dibentuk menjadi sebuah vektor sebagai input pada proses JST. Gambar 3.1 adalah langkah-langkah yang dilakukan pada tahap pra proses.

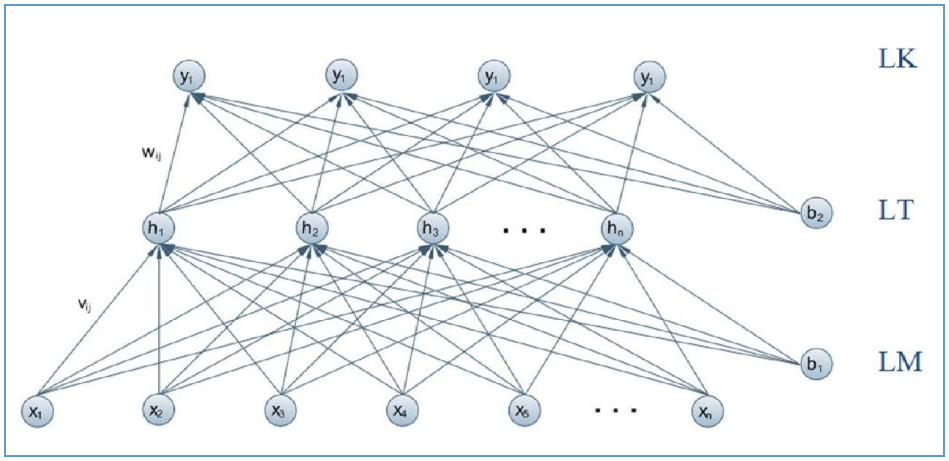


Gambar 3.1 Tahapan Metodologi

3.2 Perancangan Model JST

Dalam penelitian ini algoritma JST yang digunakan untuk mendapatkan model dan pencocokan pola adalah algoritma propagasi balik, adapun arsitektur yang digunakan adalah *Multi Layer*

Perceptron dengan satu lapisan tersembunyi seperti yang ditunjukkan pada Gambar 3.2. Agar bisa dijadikan masukan pada permodelan JST, maka setiap karakter yang telah direpresentasikan dengan matrik 10 x 8 selanjutnya diubah kedalam bentuk vektor baris ukuran 1 x 80. Sehingga terdapat sebuah matriks baru dengan ukuran 11 x 80 yang akan dijadikan masukan pada model JST.



Gambar 3.2 Model arsitektur *Multi Layer Perceptron*.

Keterangan Gambar :

LM : lapisan masukan (x_1, x_2, x_3, x_4, x_5, x_6, ..., x_n , n = 80)
LT : lapisan tersembunyi (h_1, h_2, h_3, h_4, h_5, h_6, ..., h_n , n = variasi jumlah *neurons* 9, 15, 20, 30, 40, 60)
LK : lapisan keluaran (y_1, y_2, y_3, y_4, y_5, y_6, ..., y_n , n = 11)
x_i : variabel input node i pada lapisan input, i = 0, 1, 2, …, x_n
h_j : output node j pada lapisan *hidden*, j = 0, 1, 2, …,h_n
y_k : output node k pada lapisan output, k = 1, 2, ...y_n
w_{ij} : bobot yang menghubungkan noda i pada lapisan input dengan node j pada lapisan tersembunyi
v_{jk} : bobot yang menghubungkan node j pada lapisan tersembunyi dengan node k pada lapisan output
b_1 : bias pada lapisan masukan
b_2 : bias pada layer tersembunyi

Struktur JST yang digunakan pada penelitian ini dapat dilihat pada Tabel 2

Tabel3.2 Struktur JST

Karakteristik	Spesifikasi
Arsitektur	*Multi Layer Perceptron*
Input Neurons	80
Hidden Neurons	9, 15, 20, 30, 40, 60
(Training Test)	10
Output Neurons	*Sigmoid Biner*
Fungsi aktivasi	0.1
Learning Rate (Test)	3000
Maksimum *Epoch*	

3.3 Pendefenisian Target

Dalam mendesain model JST yang dikembangkan, diharapkan dapat mengenali 10 jenis pola karakter yaitu angka 0 s/d 9. Model JST yang dibangun menggunakan fungsi aktifasi *sigmoid*

biner, fungsi aktifasi ini digunakan untuk jaringan syaraf yang dilatih dengan menggunakan metode *backpropagation* [9]. Fungsi *sigmoid biner* memiliki nilai pada range 0 sampai 1. Fungsi *sigmoid biner* dirumuskan dengan :

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

Karena model JST yang dikembangkan menggunakan fungsi aktifasi *sigmoid biner* maka diperlukan pendefenisian target sesuai dengan jumlah pola yang dinginkan, dalam hal ini terdapat 10 jenis pola yang akan dihasilkan dari desain Model JST yang dikembangkan. Ke 10 pola tersebut masing masing memiliki target berupa karakter yaitu **"0", "1", "2", "3", "4", "5", "6", "7", "8", "9"**. Pendefenisian target JST yang digunakan seperti pada Tabel 3.3.

Tabel 1.3 Definisi target

o.	Target
0	

Ket. H : Representasi angka

3.4 Metode Pelatihan dan *Testing*

Proses pelatihan dan *testing* dilakukan dalam upaya untuk mendapatkan model JST yang terbaik. Prosedur pelatihan dilakukan dengan melakukan variasi jumlah *neuron* pada lapisan tersembunyi. Tidak ada kepastian tentang berapa banyak jumlah *neuron* yang digunakan agar jaringan dapat dilatih dengan sempurna [2]. Dalam percobaan yang dilakukan arsitektur JST menggunakan variasi jumlah *neuron* pada lapisan tersembunyi adalah 9, 15, 20, 30, 40, 60 *neuron*. Kinerja dari jaringan syaraf diukur dengan melihat *error* hasil pelatihan, validasi dan *testing* terhadap sekumpulan data. Data dibagi menjadi dua bagian yang saling asing, yaitu data yang dipakai sebagai pelatihan/validasi dan data yang dipakai untuk *testing* [9]. Dalam percobaan yang dilakukan alokasi data untuk pelatihan sebayak 70% dan data untuk *testing* sebanyak 30%.

3.5 Pengambilan Keputusan

Pengambilan keputusan dilakukan dengan metode nilai maksimum untuk data yang dikenali. Jika neuron output ke-n merupakan neuron yang memiliki nilai maksimum maka neuron ke-n akan bernilai 1 (satu) dan neuron yang lain bernilai 0 (nol). Sebagai contoh, jika neuron output ketiga bernilai maksimum maka nilai output ketiga akan bernilai 1 dan neuron yang lain bernilai 0, dengan demikian sesuai tabel 3 jaringan akan mengenalinya sebagai karakter/angka "2".

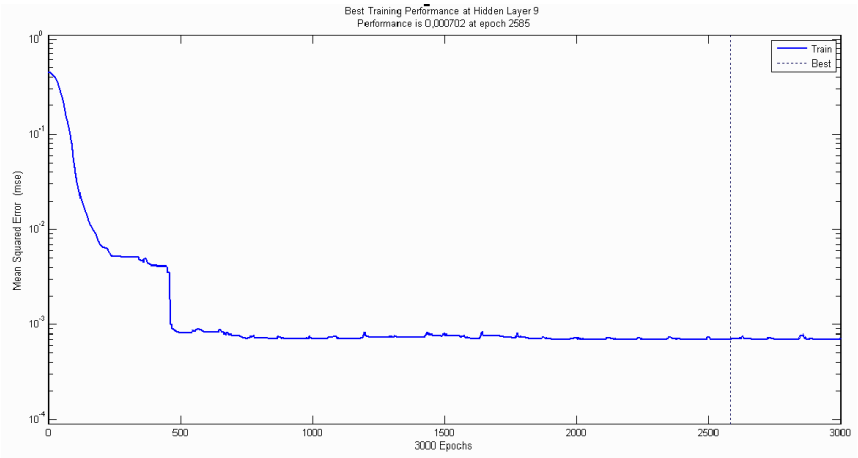
4. Pembahasan

Percobaan-percobaan dilakukan untuk mendapatkan model jaringan JST yang terbaik. Model JST terbaik adalah yang memberikan akurasi optimal ketika dilakukan validasi terhadap data *training* maupun pengujian terhadap data *testing*. Sebagai mana telah dijelaskan pada metodologi bahwa untuk proses pelatihan dengan data training menggunakan 70% dari data penelitian atau 980 data dan proses pengujian dengan data *testing* menggunakan 30% data

penelitian atau 420 data. Variasi jumlah *neuron* yang digunakan pada lapisan tersembunyi yaitu 9, 15, 20, 30, 40, dan 60 *neuron*. Metode pemberhentian iterasi dilakukan dengan menerapkan *epoch* maksimum sebanyak 3000 *epoch*. Berikut pembahasan dari setiap percobaan yang dilakukan pada masing masing variasi jumlah *neuron* lapisan tersembunyi.

4.1 Lapisan tersembunyi dengan jumlah *neuron* 9.

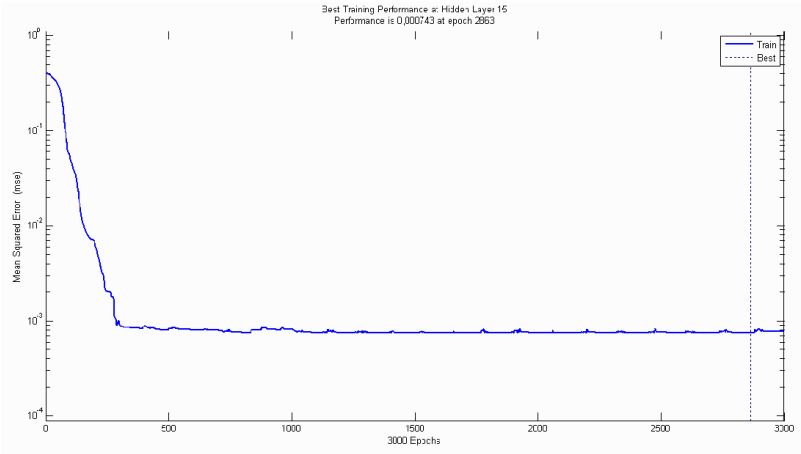
Hasil percobaan pada proses training dengan model ini dapat dilihat pada Gambar 4.1. Pada gambar tersebut ditunjukkan bahwa error terkecil yakni sebesar 0,000702 diperoleh pada *epoch* 2585 dengan durasi 33 detik. Proses validasi dilakukan dengan menguji jaringan yang terbentuk dengan data training. Hasil dari proses validasi mampu mengenali sebanyak 975 data, akurasi validasi dalam bentuk persentase adalah $\frac{975}{980} x\ 100\% = 99{,}49\%$. Pada proses pengujian dengan menggunakan data testing mampu mengenali sebanyak 408 data, akurasi pengujian dalam bentuk persentase adalah $\frac{408}{420} x\ 100\% = 97{,}14\%$.



Gambar 4.1. Pelatihan dengan 9 *neuron* pada lapisan tersembunyi.

4.2 Lapisan tersembunyi dengan jumlah *neuron* 15.

Hasil percobaan pada proses training dengan model ini dapat dilihat pada Gambar 4.2. Pada gambar tersebut ditunjukkan bahwa *error* terkecil yakni sebesar 0,000743 diperoleh pada *epoch* 2863 dengan durasi 37 detik.



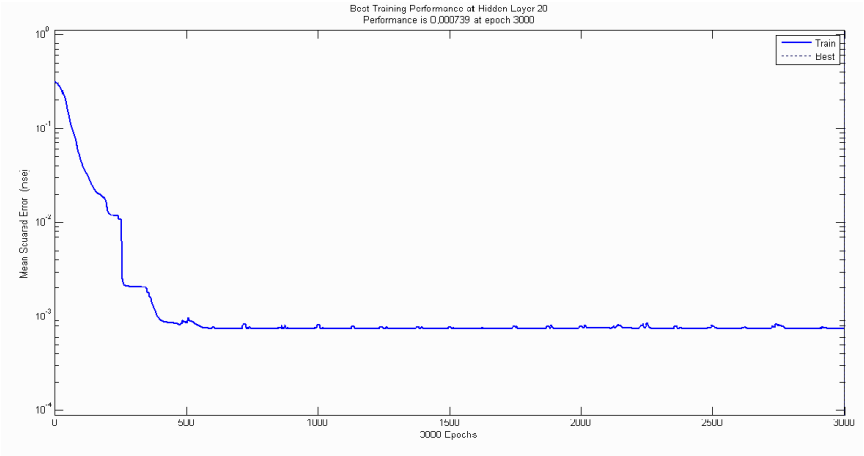
Gambar 4.2. Pelatihan dengan 15 neuron pada lapisan tersembunyi.

Proses validasi dilakukan dengan menguji jaringan yang terbentuk dengan data training. Hasil dari proses validasi mampu mengenali sebanyak 975 data, akurasi validasi dalam bentuk

persentase adalah $\frac{975}{980} x\ 100\% = 99{,}49\%$. Pada proses pengujian dengan menggunakan data testing mampu mengenali sebanyak 408 data, akurasi pengujian dalam bentuk persentase adalah $\frac{408}{420} x\ 100\% = 97{,}14\%$.

4.3 Lapisan tersembunyi dengan jumlah *neuron* 20.

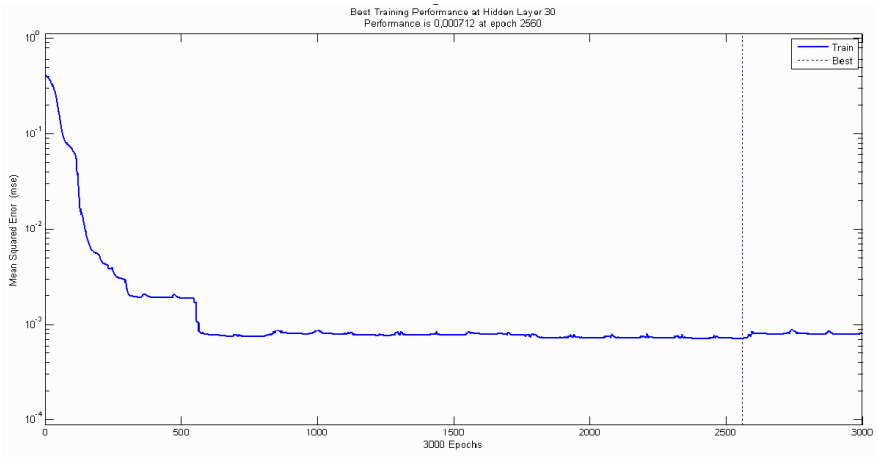
Hasil percobaan pada proses *training* dengan model ini dapat dilihat pada Gambar 4.3. Pada gambar ditunjukkan bahwa *error* terkecil yakni sebesar 0,000739 diperoleh pada epoch 3000 dengan durasi 40 detik. Proses validasi dilakukan dengan menguji jaringan yang terbentuk dengan data *training*. Hasil dari proses validasi mampu mengenali sebanyak 975 data, akurasi validasi dalam bentuk persentase adalah $\frac{975}{980} x\ 100\% = 99{,}49\%$. Pada proses pengujian dengan menggunakan data *testing* mampu mengenali sebanyak 408 data, akurasi pengujian dalam bentuk persentase adalah $\frac{408}{420} x\ 100\% = 97{,}14\%$.



Gambar 4.3 Pelatihan dengan 20 neuron pada lapisan tersembunyi.

4.4 Lapisan tersembunyi dengan jumlah *neuron* 30.

Hasil percobaan pada proses *training* dengan model ini dapat dilihat pada Gambar 4.4. Pada gambar ditunjukkan bahwa *error* terkecil yakni sebesar 0,000712 diperoleh pada *epoch* 2560 dengan durasi 46 detik.



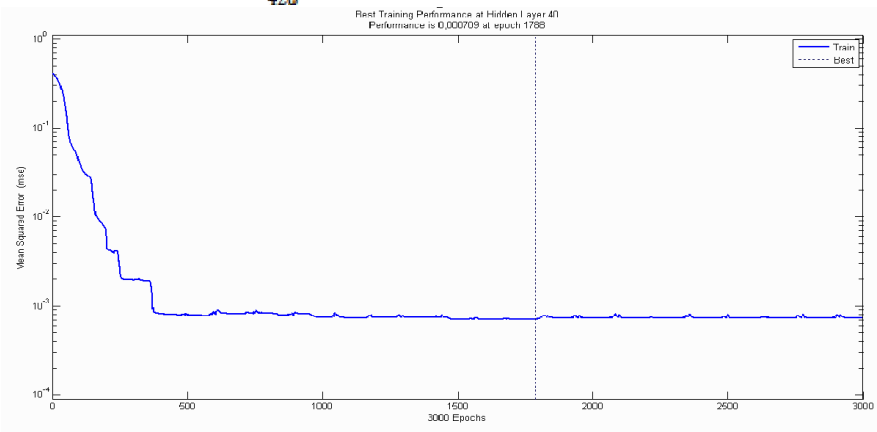
Gambar 3. Pelatihan dengan 30 neuron pada lapisan tersembunyi.

Proses validasi dilakukan dengan menguji jaringan yang terbentuk dengan data *training*. Hasil dari proses validasi mampu mengenali sebanyak 975 data, akurasi validasi dalam bentuk

persentase adalah $\frac{975}{980} x\ 100\% = 99{,}49\%$. Pada proses pengujian dengan menggunakan data testing mampu mengenali sebanyak 410 data, akurasi pengujian dalam bentuk persentase adalah $\frac{410}{420} x\ 100\% = 97{,}62\%$.

4.5 Lapisan tersembunyi dengan jumlah *neuron* 40.

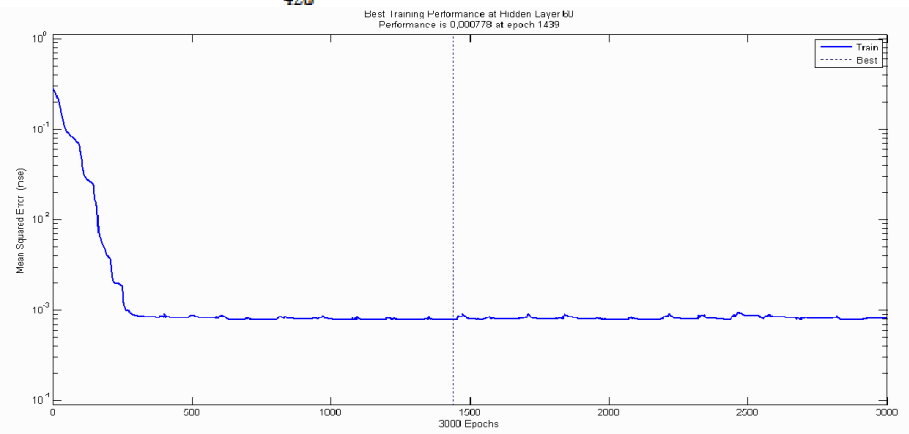
Hasil percobaan pada proses *training* dengan model ini dapat dilihat pada Gambar 4.5. Pada gambar ditunjukkan bahwa error terkecil yakni sebesar 0,000709 diperoleh pada *epoch* 1788 dengan durasi 57 detik. Proses validasi dilakukan dengan menguji jaringan yang terbentuk dengan data training. Hasil dari proses validasi mampu mengenali sebanyak 975 data, akurasi validasi dalam bentuk persentase adalah $\frac{975}{980} \boldsymbol{x\ 100\% = 99{,}49\%}$. Pada proses pengujian dengan menggunakan data *testing* mampu mengenali sebanyak 410 data, akurasi pengujian dalam bentuk persentase adalah $\frac{410}{420} \boldsymbol{x\ 100\% = 97{,}62\%}$.



Gambar 4.5 Pelatihan dengan 40 *neuron* pada lapisan tersembunyi.

4.6 Lapisan tersembunyi dengan jumlah *neuron* 60.

Hasil percobaan pada proses *training* dengan model ini dapat dilihat pada Gambar 4.6. Pada gambar ditunjukkan bahwa *error* terkecil yakni sebesar 0,000778 diperoleh pada *epoch* 1439 dengan durasi 72 detik. Proses validasi dilakukan dengan menguji jaringan yang terbentuk dengan data *training*. Hasil dari proses validasi mampu mengenali sebanyak 975 data, akurasi validasi dalam bentuk persentase adalah $\frac{975}{980} \boldsymbol{x\ 100\% = 99{,}49\%}$. Pada proses pengujian dengan menggunakan data *testing* mampu mengenali sebanyak 409 data, akurasi pengujian dalam bentuk persentase adalah $\frac{409}{420} \boldsymbol{x\ 100\% = 97{,}38\%}$.



Gambar 4. Pelatihan dengan 60 *neuron* pada lapisan tersembunyi.

Ringkasan hasil dari percobaan percobaan yang dilakukan dengan variasi jumlah *neuron* pada lapisan tersembunyi dapat dilihat pada Tabel 4.6

Tabel 2.6 Tabel hasil pelatihan/validasi dan testing

Lapisan Tersembunyi	Durasi Pelatihan	MSE	Epoch	Akurasi Validasi	Akurasi *Testing*
9 *Neurons*	33 detik	0,000699	2914	99,49%	97,14%
15 *Neurons*	37 detik	0,000743	2863	99,49%	97,14%
20 *Neurons*	40 detik	0,000739	3000	99,49%	97,14%
30 *Neurons*	46 detik	0,000712	2560	99,49%	97,62%
40 *Neurons*	57 detik	0,000709	1788	99,49%	97,62%
60 *Neurons*	72 detik	0,000778	1439	99,49%	97,38%

Dari Tabel, terlihat bahwa Model JST dengan variasi *neuron* pada lapisan tersembunyi yaitu 30 dan 40 memberikan akurasi testing terbaik yakni sebesar 97,62%. Hal ini menunjukkan bahwa pada model-model tersebut kinerja jaringan lebih stabil, dan akan memberikan akurasi pengenalan lebih baik.

5. Keseimpulan

Dari penelitian yang telah dilakukan dengan hasil yang cukup memuaskan, maka kesimpulan yang dapat diberikan adalah, model JST yang dikembangkan untuk mendapatkan pola pengenalan angka 0 s/d 9 yang paling baik adalah yang menggunakan 80 lapisan masukan, 30 lapisan tersembunyi dan 10 lapisan keluaran karna pola tersebut mampu menghasilkan akurasi validasi 99,49% dan akurasi pengujian 97,62%.

Daftar Pustaka

[1] Park SS, Jung WG, Shin YG, Jang DS. 2008. *Optical Character Recognition System Using BP Algorithm*. IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.12.

[2] Siang JJ. 2009. *Jaringan Syaraf Tiruan & Pemrogramannya Menggunakan MATLAB.* Yogyakarta : Andi Yogyakarta.

[3] Purnomo MH, Muntasa A. 2010. *Konsep Pengolahan Citra Digital dan Ekstraksi Fitur*. Yogyakarta : Graha Ilmu.

[4] Haryono MEA. 2005 . *Pengenalan Huruf Menggunakan Model Jaringan Syaraf Tiruan Radial Basis Function dengan Randomize Cluster Decision*. Didalam : Seminar Nasional Aplikasi Teknologi Informasi 2005 (SNATI 2005) : Yogyakarta, 18 Juni 2005.

[5] Sholahuddin A. 2002 . *Penerapan neural network tentang metode backpropagation pada pengenalan pola huruf* . Didalam : Komputer dan Sistem Intelijen (KOMMIT2002) Auditorium Universitas Gunadarma : Jakarta , 21 – 22 Agustus 2002.

[6] Saputro N 2003. *Pengenalan Huruf dengan memakai Algoritma Genetik*. INTEGRAL, Vol. 8 No. 2. Universitas Katolik Parahyangan: Bandung.

[7] Wijaya MC, Prijono A. 2007. *Pengelolaan Citra Digital Menggunakan MatLAB*. Bandung : Penerbit Informatika.

[8] Wirayuda TAB, Vaulin S, Dayawati RN. 2009. *Pengenalan Huruf Komputer Menggunakan Algoritma berbasis Chain Code dan Algoritma Sequence Alignment*. Didalam: Konferensi Nasional Sistem dan Informatika 2009 : Bali, November 14, 2009.

[9] Kusumadewi S. 2004. *Membangun Jaringan Syaraf Tiruan Menggunakan MATLAB & EXCEL LINK*. Yogyakarta. Graha Ilmu.

[10] Basuki A, Palandi JF, Fatchurrochman. 2005. *Pengolahan Citra Digital mengunakan Visual Basic*. Yogyakarta. Graha Ilmu.

[11] Putra D. 2010. *Pengeloaan Citra Digital*. Yogyakarta : Andi Yogyakarta.

[12] Puspitanigrum D. 2006. *Pengatar Jaringan Syaraf Tiruan*. Yogyakarta : Andi Yogyakarta.