PEMODELAN *DOWNSAMPLING* UNTUK OPTIMASI PENGENALAN POLA JARINGAN SARAF TIRUAN (STUDI KASUS: LEMBAR JAWABAN UJIAN PERGURUAN TINGGI)

KANI

SEKOLAH PASCASARJANA
INSTITUT PERTANIAN BOGOR
BOGOR
2017

PEMODELAN *DOWNSAMPLING* UNTUK OPTIMASI PENGENALAN POLA JARINGAN SARAF TIRUAN (STUDI KASUS: LEMBAR JAWABAN UJIAN PERGURUAN TINGGI)

K A N I

SEKOLAH PASCASARJANA
INSTITUT PERTANIAN BOGOR
BOGOR
2017

PERNYATAAN MENGENAI TESIS DAN SUMBER INFORMASI SERTA PELIMPAHAN HAK CIPTA

Dengan ini saya menyatakan bahwa tesis berjudul Pemodelan *Downsampling* untuk Optimasi Pengenalan Pola Jaringan Saraf Tiruan (Studi Kasus: Lembar Jawaban Ujian Perguruan Tinggi) adalah benar karya saya dengan arahan dari komisi pembimbing dan belum diajukan dalam bentuk apa pun kepada perguruan tinggi mana pun. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan maupun tidak diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam Daftar Pustaka di bagian akhir disertasi ini.

Dengan ini saya melimpahkan hak cipta dari karya tulis saya kepada Institut Pertanian Bogor.

Bogor, April 2017

Kani
NIM G651150101

RINGKASAN

KANI. Pemodelan *Downsampling* untuk Optimasi Pengenalan Pola Jaringan Saraf Tiruan (Studi Kasus: Lembar Jawaban Ujian Perguruan Tinggi). Dibimbing oleh IRMAN HERMADI dan AGUS BUONO.

Saat ini kemajuan teknologi digital telah mempercepat transformasi di bidang teknologi informasi memberikan inspirasi positif bagi para ilmuwan. Baru-baru ini dalam aplikasi Jaringan Saraf Tiruan (JST) dari pengenalan pola, telah digunakan untuk pengenalan pola tulisan tangan, namun, metode ini belum diterapkan di digunakan dalam mengenali huruf pada lembar jawaban ujian. Oleh karena itu, studi bertujuan memodelkan praproses untuk pengurangan pixel menggunakan teknik *downsampling* dengan harapan mengoptimalkan pengenalan pola pada Jaringan Saraf Tiruan (JST).

Penelitian ini mengambil studi kasus pola tulisan tangan untuk tulisan tangan huruf 'A', 'B', 'C', 'D' dan 'E'. Teknik Model *downsampling* yang diusulkan adalah dengan mereduksi matriks citra pada kolom dan baris sehingga menghasilkan vektor sebagai *input* JST.

Hasil percobaan, menunjukkan bahwa ketika kedua kolom dan baris matriks yang direduksi mendapatkan akurasi yang lebih tinggi (98,8%) dengan rentang rendah (3.30%) dan konvergensi waktu yang rendah (9 menit dan 20 detik) bila dibandingkan dengan *Backpropagation* normal, sehingga metode yang diusulkan disajikan keandalan yang lebih tinggi untuk pengenalan pola tulisan tangan dengan RMSE rata-rata 0,1.

Kata kunci: *Backpropagation*, *Downsampling*, JST.

SUMMARY

KANI. Modeling Optimization Downsampling for Pattern Recognition Artificial Neural Network (Case Study: Higher Education Exam Answer Sheet). Supervised by IRMAN HERMADI and AGUS BUONO.

Currently advancement in digital technology has accelerated transformation in the field of information technology giving a positive inspiration for scientific communication. Recently application artificial neural network (ANN) in pattern recognition, have been used for handwriting pattern recognition, however, this method has not been applied in used in collage to recognize letters in examination answer sheet. Therefore, the current study purposed image prepressing model for pixel reduction using downsampling technique with an aim to optimize the basic artificial neural network (ANN).

Therefore, the current research used considered a case study of handwritten pattern for exam letters 'A', 'B', 'C', 'D' and 'E'. The proposed model downsampling technique was first applied to rescale the image pixel matrix on both columns and rows after which the resultant matrix acted as input vector for ANN.

The results of the simulation, demonstrated that when both column and row matrixes are rescaled using the downsampling higher accuracy (98.8%) was recorded with low range (3.30%) and low convergence time (9mint & 20 sec) when compared with normal Back-propagation, thus the proposed method presented higher reliability for handwriting pattern recognition with an average RMSE of 0.1.

Keywords: ANN, Back-propagation, Downsampling

PEMODELAN *DOWNSAMPLING* UNTUK OPTIMASI PENGENALAN POLA JARINGAN SARAF TIRUAN (STUDI KASUS: LEMBAR JAWABAN UJIAN PERGURUAN TINGGI)

K A N I

Tesis
sebagai salah satu syarat untuk memperoleh gelar
Magister Komputer
pada
Program Studi Ilmu Komputer

SEKOLAH PASCASARJANA
INSTITUT PERTANIAN BOGOR
BOGOR
2017

Penguji Luar Komisi pada Ujian Tesis: Dr Eng Wisnu Ananta, ST MT.

Judul Tesis : Pemodelan *Downsampling* untuk Optimasi Pengenalan Pola Jaringan Saraf Tiruan (Studi Kasus: Lembar Jawaban Ujian Perguruan Tinggi)
Nama : Kani
NIM : G651150101

Disetujui oleh

Komisi Pembimbing

Irman Hermadi, SKom MS PhD
Ketua

Dr Ir Agus Buono, MSi MKom
Anggota

Diketahui oleh

Ketua Program Studi
Ilmu Komputer

Dr Ir Sri Wahjuni, MT

Dekan Sekolah Pascasarjana

Dr Ir Dahrul Syah, MSc Agr

Tanggal Ujian: 24 Maret 2017

Tanggal Lulus: 04 APR 2017

PRAKATA

Puji dan syukur penulis panjatkan kehadirat Allah *subhanahu wa ta'ala* atas segala karunia-Nya sehingga karya ilmiah ini berhasil diselesaikan. Tema yang dipilih dalam penelitian yang dilaksanakan sejak bulan Juli 2016 ini adalah kecerdasan buatan dengan judul Pemodelan *Downsampling* untuk optimasi Pengenalan Pola Jaraingan Saraf Tiruan dengan Studi Kasus: Lembar Jawaban Ujian (LJU) Perguruan Tinggi.

Terima kasih penulis ucapkan kepada Bapak Irman Hermadi, Ph.D selaku pembimbing I dan Bapak Dr. Agus Buono selaku pembimbing II. Di samping itu, ucapan terima kasih Dekan FMIPA IPB Ibu Dr. Sri Nurdyati, Ketua Program Studi Ilmu Komputer FMIPA IPB Ibu Sri Wahjuni, MT. penghargaan penulis sampaikan kepada Ayahanda H. Launggu (almarhum) dan Ibunda Hj. Tika (almarhumah), serta seluruh keluarga, atas segala doa, kasih sayangnya dan semangat yang selalu diberikan kepada saya selama menempuh perkuliahan. Tidak lupa juga kepada Pimpinan Universitas Terbuka Ibu Rektor Prof. Tian Belawati, Pembatu Rektor I : Ibu Dr. Dewi Padmo, Pembantun Rektor II : Ibu Dr. Yuni Tri Hewindati, Pembantu Rektor III : Bapak Dr. Amin Zuhairi, dan Pembantu Rektor IV : Bapak Dr. Mohammad Yunus. Tak lupa pula jajaran LPBAUSI, Bapak Prof. Dr. Odjat Darojat selaku Kepala LPBAUSI, Kepala Puskom Dyah Paminta Rahayu (Alumni Magister Ilmu Komputer IPB), dan ucapan terima kasih terkhusus buat Bapak Arga yang telah banyak memberikan semangat dan motivasi tinggi kepada Saya selama pra kuliah maupun selama kuliah, begitu juga dengan Bapak Unggul Utan Sufandi sebagai alumni IPB Magister Ilmu Komputer angkatan pertama yang telah banyak memberikan masukan, bimbingan dan perbandingan ilmu yang sangat baik.

Penghargaan penulis saya sampaikan kepada:

1. Keluarga kecil saya, Isteri saye Nurdjawahir Achma Asiri, Anak-anak saya Zahra Nur Aathiroh Putri Kani, Zahran Nur Azizan Putra Kani dan Zahira Nur Assyifa Putri Kani, serta mertua.
2. Rekan-rekan satu bimbingan di Laboratorium SEIS yaitu Pak Abdul Kadir Salam, Pak Arham Amri, Pak Andri Sungkono, Dyta Kameswara (Kodok), Ella Macheta, Intan Purnama Sari.
3. Rekan-rekan Anak bimbingan Pak Agus Buono, yaitu: Pak Hastuadi (memberikan pencerahan tentang ilmu pengenalan pola), Angga Pratama (Statistik) dan Agus Arbawi.
4. Anak bimbingan Bu Imas, yaitu: Apriliantono dan kawan-kawan.
5. Ucapan terima kasih kepada teman mahasiswa internasional Peter Juma Ocheng, yang banyak memberikan masukan dan saran dalam penelitian ini.
6. Rekan-rekan Angkatan 2015 dan 2014 yang saya tidak bisa sebut satu persatu

Semoga karya ilmiah ini bermanfaat.

Bogor, April 2017

Kani

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR GAMBAR

1 PENDAHULUAN

Latar Belakang

Revolusi digital yang dimulai dari tahun 1980 manandai sebuah pergeseran teknologi mekanik dan teknologi analog menjadi teknologi digital, era digital telah merubah pola pikir manusia menuju pola praktis dan efisien. Revolusi digital telah memberikan efek signifikan dalam pola kerja saat ini, peralihan kata "manual" ke "automatisasi" menjadi sangat mungkin dengan perkembangan teknologi informasi. Kemampuan ilmuan untuk mengkolaborasi metode ilmuan terdahulu dengan sentuhan teknologi informasi menjadi sebuah metode baru adalah sebuah kecemerlangan yang menjadi warna tersendiri bagi khazanah ilmu pengetahuan, salah satu ilmu pengetahuan yang berkembang saat ini adalah pengenalan pola (*pattern recognition*). Tidak bisa dibantah bahwa pengenalan pola telah memberikan solusi untuk berbagai masalah mulai dari pengenalan suara, pengenalan wajah, klasifikasi tulisan tangan dan diagnosa medis (Asht *et al.* 2012).

Pengenalan pola adalah disiplin ilmu yang tujuannya adalah mengklasifikasi objek menjadi beberapa kategori atau kelas (Theodoridis *et al.* 2003). Pengenalan pola adalah studi tentang bagaimana mesin dapat mengamati lingkungannya, belajar membedakan objek dari latar belakangnya (Basu *et al.* 2010) atau pengenalan pola dapat diartikan sebagai sebuah proses untuk mengenali objek secara otomatis berdasarkan ciri yang didasarkan pada data pelatihan (Muntasa 2015). Salah satu metode pengenalan pola yang paling terkenal adalah Jaringan Saraf Tiruan (*JST*). Tahun 2011 UT telah melakukan uji coba pengenalan huruf tulisan tangan dengan JST dan dalam penelitian tersebut disimpulkan bahwa JST sangat berpotensi digunakan untuk mengenali huruf besar tulisan tangan (Dwi *et al.* 2011). Hasil percobaan menunjukkan bahwa JST yang dioptimalkan menurunkan jumlah iterasi dibanding dengan JST tradisional pada proses pembelajaran, pelatihan dan akurasi cukup tinggi (Chen *et al.* 2013). Teknologi pengenalan pola telah banyak digunakan oleh masyarakat, swasta, pemerintahan, militer, perguruan tinggi baik untuk melayani masyarakat maupun untuk kebutuhan internal.

Ilmu untuk mengenali karakter tulisan tangan disebut dengan *handwriting recognition*. *Handwriting recognition* telah menjadi salah satu bidang penelitian yang paling menarik dan menantang di bidang pengolahan citra dan pengenalan pola pada beberapa tahun terakhir (J.Pradeep *et al.* 2011). Pemanfaatan ilmu pengenalan pola juga telah merambah dunia pendidikan, misalnya perguruan tinggi, contoh Universitas Terbuka (UT) sekaligus menjadi studi kasus dalam penelitian ini, UT diawal berdirinya telah menggunakan teknologi pendidikan untuk pembelajaran mahasiswa jarak jauh, menggunakan *digital printing* untuk pencetakan bahan ujian, dan menggunakan algoritme pengenalan pola dalam memeriksa hasil ujian yang ada pada Lembar Jawaban Ujian (LJU).

Khusus pada LJU, UT pada setiap masa ujian memproses sekitar 1 800 000 LJU program Pendidikan Dasar (Pendas), dan 350 000 LJU program Non Pendidikan Dasar (Non Pendas) melalui unit Pusat Pengujian (PUSJIAN), hal yang cukup serius pada proses diatas adalah begitu banyak LJU berkasus, LJU terdeteksi salah begitu diperiksa. Dan secara umum terdapat dua jenis kasus LJU, yaitu karena proses *scanning*/digitasi yang tidak sempurna dan kesalahan mahasiswa dalam

mengisi LJU (dwi *et al.* 2011). Proses *scanning*/digitasi yang tidak sempurna dapat disebabkan oleh mesin *scanner* yang tidak berfungsi dengan sempurna atau kualitas/kondisi LJU yang kurang baik. Kesalahan mahasiswa dalam mengisi LJU biasanya karena mahasiswa tidak menandatangani LJU atau tanda tangan di LJU tidak sama dengan tanda tangan di daftar hadir ujian, dan kesalahan dalam mengisi identitas utama pada. Kesalahan-kesalahan seperti ini tidak bisa ditolelir, akibatnya nilai mahasiswa tidak bisa diumumkan tepat waktu. Selain merugikan mahasiswa, di samping itu kasus-kasus LJU ini dapat menghambat proses pengolahan LJU dan menambah beban operasional, dampak psikologisnya adalah adalah dapat menurunkan tingkat kepercayaan masyarakat terhadap UT sendiri.

Permasalahan berikutnya adalah pada pemeriksaan LJU, dalam prosesnya pemeriksaan banyak ditemukan kesalahan mahasiswa dalam menghitamkan bulatan pada LJU. Contoh data masa registrasi 2015.2 (satu semester, dari bulan Juli sampai bulan Desember), terdapat 32 277 kasus Program Pendas dan 6 878 kasus untuk Program Non Pendas (Data Pusat Pengujian UT), hal ini bisa dimaklumi karena 75,95% mahasiswa UT berusia diatas 30 tahun bukan *fresh graduate* sehingga tidak terbiasa menggunakan LJU (dwi *et al.* 2011). Kesalahan seperti ini berdampak signifikan, yaitu tidak terbacanya LJU dengan sempurna yang berdampak kepada nilai UAS yang menentukan lulus tidaknya mahasiswa dalam suatu matakuliah. Permasalahan lain yang dihadapi adalah adanya kekhawatiran pada diri Mahasiswa yang memiliki keraguan dalam melakukan penghitaman bulatan yang menyebabkan kesalahan pembacaan jawaban oleh peralatan *scanner* yang dimiliki oleh UT, Dari permasalahan-permasalahan yang telah disebutkan di atas, maka perlu mencari solusi alternatif untuk menangani pemeriksaan hasil ujian tersebut dengan Pengenalan Pola (*pattern recognition*) yang lebih layak dan teroptimasi.

Perlu pengujian lebih lanjut untuk JST yang dioptimasi untuk meningkatkan akurasi pengenalan (Perwej *at al.* 2011), Salah cara untuk meningkatkan akurasi pengenalan pola adalah dengan melakukan optimasi pada bagian ektraksi ciri. Beberapa pertanyaan berikut menjadi titik awal untuk mencari model ektraksi ciri adalah; Apakah metode ektraksi ciri harus berdasarkan hitungan matematika secara logis? Apakah mungkin merekontruksi dan meminimumkan ukuran matriks citra dari aslinya dengan cara reduksi relevan untuk digunakan sebagai input pada JST untuk pengenalan pola? Pertanyaan ini dijawab oleh penelitian Sarangi *et al.* (2014) bahwa sebuah ektraksi ciri boleh dengan menemukan sebuah set yang lebih kecil yang mewakili objek asli tanpa mengurangi nilai atau makna dari citra aslinya. Pada percobaan tersebut merubah nilai digitasi menjadi nilai desimal.

Sangat minimnya literatur yang bisa memberikan jawaban yang memuaskan atas pertanyaan-pertanyaan diatas yang memotivasi penulis untuk mengusulkan sebuah pemodelan metode ektraksi ciri dengan *downsampling* (reduksi) baris citra untuk mengoptimasi pengenalan pola JST dengan mengkolaborasi latar belakang permasalahan UT pada pengenalan pola huruf tulisan tangan untuk LJU. Sebagai pembanding, Dwi *et al.* (2011) mengatakan bahwa "Metode JST-BP memberikan hasil dengan tingkat presisinya baru mencapai 87,35%. Idealnya untuk dapat digunakan mengolah data ujian presisi harus mencapai 100%", penelitian tersebut menggunakan JST tanpa optimasi dan penelitian ini mengikuti saran yang ada pada penelitian tersebut disamping motivasi yang telah disebutkan di atas.

Perumusan Masalah

Pertanyaan yang muncul pada latar belakang yaitu: Apakah metode ektraksi ciri harus berdasarkan hitungan matematika secara logis? Apakah mungkin merekontruksi dan meminimunkan ukuran matriks citra dari aslinya dengan cara reduksi relevan untuk digunakan sebagai input pada JST untuk pengenalan pola?, kemudian dipadu dengan latar belakang permasalahn yang dihadapi oleh UT, meskipun dengan menggunakan sistem LJU dengan cara menghitamkan bulatan yang sudah tersedia telah berkontribusi banyak dalam pemeriksaan hasil ujian, akan tetapi tidak terlepas dari banyak kesalahan. Untuk mengurangi kesalahan-kesalahan dalam menghitamkan bulatan yang ada pada LJU, telah dikembangkan LJU dengan menggunakan tulisan tangan dan pengenalan menggunakan Metode JST, dengan tidak teroptimasi, namun tingkat hasil presisinya masih rendah (Dwi *et al.* 2011), maka penelitian ini mengajukan solusi optimasi pada ekstraksi fitur atau ektraksi ciri, Ektraksi fitur atau ekstraksi ciri adalah proses menemukan satu set yang lebih kecil dari elemen yang mewakili objek asli sehingga percobaan komputerisasi bisa dicapai lebih cepat dan dengan penggunaan memori yang lebih sedikit (Pradeepta *et al.* 2014). Penelitian ini mengajukan sebuah Pemodelan *downsampling* (reduksi) matriks citra untuk mendapatkan satu set ektraksi ciri untuk optimasi pengelanan pola JST sehingga menghasilkan akurasi yang tinggi dan waktu yang relatif cepat pada LJU.

Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah mengembangkan sebuah model pada ekstrkasi ciri dengan cara *downsampling* matriks citra untuk optimasi pengenalan pola JST dalam mengenali tulisan tangan pada LJU.

Manfaat Penelitian

Pengenalan pola JST yang dioptimasi dengan model *downsampling* untuk pengenalan tulisan tangan ini dapat digunakan untuk mengenali huruf kapital tulisan tangan pada LJU oleh Pendidikan Tinggi yang memanfaatkan ujian dengan sistem *multiple choice* untuk melakukan seleksi dan evaluasi.

Ruang Lingkup Penelitian

Melihat begitu banyaknya potensi yang akan mungkin bisa dikembangkan lebih lanjut dalam penelitian ini maka dibatasi ruang lingkupnya hanya pada menghasilkan pemodelan ektraksi ciri untuk optimasi JST dalam mengenali pola huruf kapital 'A', 'B', 'C', 'D', dan 'E'.

2 TINJAUAN PUSTAKA

Pengolahan Citra

Citra dan Pengolahan Citra

Citra adalah suatu representasi, kemiripan, atau imitasi dari suatu objek atau benda (Achmad *et al.* 2015). Citra digital adalah representasi dari sebuah gambar dua dimensi sebagai sebuah kumpulan nilai digital yang disebut elemen gambar atau piksel (Bhat 2014). Pengolahan citra sebuah metode untuk melakukan beberapa operasi pada citra, untuk mendapatkan citra yang disempurnakan atau untuk mengambil beberapa informasi yang berguna dari citra. *Input* dari sebuah pengolahan citra adalah citra itu sendiri sedangkan keluarannya adalah dapat berupa citra atau sekumpulan parameter yang berhubungan dengan citra itu sendiri.

Histogram

Histogram adalah suatu grafik yang menggambarkan distribusi data dari derajat keabuan dalam sebuah citra digital (Saravanan *et al.* 2014). Histogram adalah sebuah metode yang digunakan untuk menggambarkan distribusi frekuensi sinyal digital yang terdiri dari beberapa pemetaan (data *binning*), dengan masing-masing data *bin* sesuai dengan rentang nilai (Yoo *et al.* 2006). Histogram digunakan dalam pekerjaan pengolahan citra untuk mengetahui kualitatif dan kuantitaf dari sebuah citra, Histogram dapat memberikan informasi dari citra, misalnya : kecerahan, kontras, derajat keabuan (*gray-level histogram*), pengambangan (*thresholding*) dan sebagai alat bantu untuk dapat menganalisa tingkat kualitas dan kuantitas citra. Histogram tingkat keabuan adalah suatu fungsi yang menunjukkan jumlah titik yang ada di dalam suatu citra untuk setiap tingkat keabuan (Achmad *et al.* 2006). Histogram adalah sebuah *tool* yang sering digunakan pengolahan citra untuk dengan tugas pengambilan citra, klasifikasi citra, bentuk pencocokan, dan pengenalan objek (Yang *et al.* 2014). Ada 2 manfaat utama dari histogram yaitu : Penentuan parameter digitisasi, digunakan sebagai indikasi visual untuk menentukan apakah suatu citra berada pada jangkauan yang tepat dalam skala keabuan. Pemilihan batas ambang, teknik pemilihan batas ambang biasa disebut dengan proses *thresholding*.

Thresholding

Thresholding adalah teknik yang penting dalam aplikasi segmentasi citra (Vala *et al.* 2013). Ide dasar dari *thresholding* adalah untuk memilih nilai ambang *gray-level* yang optimal untuk memisahkan objek citra (Achmad *et al.* 2005), contoh Gambar 3, citra yang sudah repsentasikan kedalam histogram dengan dua puncak, menunjukkan latar belakang gelap dan dan objek berwarna terang.

Thresholding dalam pengolahan citra dibagi menjadi dua, yaitu : *Thresholding* Global dan *Thresholding* Lokal. *Thresholding* Global adalah jika nilai T tergantung hanya pada $f(x,y)$ (Vala *et al.* 2013) saja contoh metode ini adalah metode Otsu, sedangkan *thresholding* lokal adalah jika nilai T tergantung $f(x,y)$

dan $p(x,y)$) (Vala *et al.* 2013) dimana $p(x,y)$ merupakan ciri khusus yang miliki setiap piksel, salah satu ciri khusus adalah nilai rerata histogram tingkat keabuan (*gray-level histogram*) piksel-piksel tetangga yang berpusat di (x,y). $g(x,y)$ adalah nilai piksel hasil *threshold* pada baris x dan kolom y. Dan $f(x,y)$ adalah nilai piksel dari citra *input* asal pada baris x dan kolom y dan T adalah nilai *threshold*.

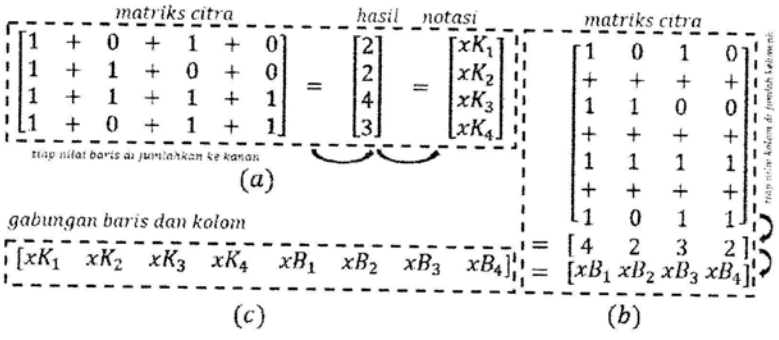
Kelebihan dan kekurangan diantara keduanya, untuk *Thresholding* Global : Proses hingga semua bagian citra selesai dengan cepat, jika iluminasi (tingkat terang) pada citra merata akan memberikan hasil yang bagus, tidak berhasil/gagal jika ada perubahan iluminasi pada bagian-bagian tertentu. Sedangkan *Thresholding* Lokal : Proses lebih lambat, beradaptasi dengan perubahan iluminasi (tingkat terang), tingkat keberhasilnnya akan sangat baik pada iluminasi yang *uniform* (merata) ataupun tidak.

Metode Otsu

Otsu *thresholding* merupakan suatu metode yang membantu untuk melakukan segmentasi citra digital dengan cara mencari *threshold* menggunakan analisa statistik histogram (Saravanan *et al.* 2014), disinilah *thresholding* memainkan peran utamanya dalam *binarization* citra. Citra *binarization* adalah proses pemisahan nilai-nilai piksel menjadi dua kelompok, putih sebagai latar belakang dan hitam sebagai objek citra (Bhargava *et al.* 2014), tujuan dari metode otsu adalah membagi histogram citra *gray level* ke dalam dua daerah yang berbeda secara otomatis untuk memasukkan nilai ambang. Pendekatan yang dilakukan oleh metode otsu adalah dengan melakukan analisis diskriminan yaitu menentukan suatu variabel yang dapat membedakan antara dua atau lebih kelompok yang muncul secara alami

Dowsampling (Reduksi)

Ektraksi Fitur atau ekstraksi ciri adalah proses menemukan satu set yang lebih kecil dari elemen yang mewakili objek asli sehingga percobaan komputerisasi bisa dicapai lebih cepat dan dengan penggunaan memori yang lebih sedikit (Sarangi *et al.* 2014). Proses *Downsampling (reduction)* adalah proses pereduksian nilai-nilai matriks citra *mxn* menjadi nilai baru dengan cara men-*decode* nilai matriks



Gambar 1 (a) Penjumlahan nilai-nilai kolom pada tiap baris, (b) Penjumlahan nilai-nilai baris pada tiap kolom, (c) Penjumlahan keduanya (kolom dan baris)

citra perbaris atau matriks citra perkolom atau keduanya menjadi nilai baru atau matriks baru $m \times 1$ atau $1 \times n$ dan kemudian diubah menjadi vektor kolom. Dengan tereduksinya nilai-nilai matriks asli menjadi matriks baru dan mengurangi jumlah *input* untuk JST. Syarat yang harus di penuhi matriks citra ialah nilai-nilai matriks citra harus nilai matriksnya adalah 0 atau 1.

Perhatikan Gambar 4, untuk Gambar 4.a mempunyai 4 *input* nilai baru (xK_1, xK_2, xK_3, xK_4), Gambar 4.b mempunyai 3 nilai *input* baru (xB_1, xB_2, xB_3, xB_3), dan Gambar 4.c mempunyai 6 *input* baru (xK_1, xK_2, xK_3, xK_4, xB_1, xB_2, xB_3, xB_3).

Misalnya sebuah citra gambar dengan dimensi matriks 10x10, normalnya data *input* yang dihasilkan dari matriks tersebut adalah vektor 100 kolom, akan tetapi dengan *downsampling*, hanya akan menghasilkan vektor 10 kolom (kolom matriks) atau vektor 10 kolom (baris matriks) atau vektor 20 kolom saja (*10 kolom + 10 baris*), sebagai contoh matriks 10x10 berikut ini:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Matriks citra di atas jika dilakukan *downsampling* (reduksi) maka didapatkan sebagai berikut, jika mereduksi kolom maka vektor 10 kolom (sisi kolom) atau jika mereduksi baris maka dapat vektor 10 kolom, jika digabung maka vektor 20 kolom (10+10).

Downsampling (Reduksi) Kolom (DS-K) Matriks

Downsampling (Reduksi) kolom DS-K) Matriks adalah menjadikan kolom menjadi pondasi perhitungan, semua nilai-nilai matriks pada kolom dalam tiap baris dijumlahkan dari kiri ke kenan sehingga menghasilkan matriks baru *m x 1* atau vektor dengan *m* kolom.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad direduksi(xK) = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 2 \\ 2 \\ 6 \\ 2 \\ 2 \\ 2 \\ 0 \end{bmatrix}$$

Hasil *dowsampling* (reduksi) dari matriks citra 10x10 untuk kolom adalah matriks 10x1 dan jika divektorkan menghasilkan vektor 10 kolom. Kalau diformulasikan secara umum untuk sebuah penjumlahan nilai-nilai kolom pada tiap baris matriks citra sebagai berikut :

$$\begin{bmatrix} x_{11} + x_{12} + x_{13} + x_{14} \\ x_{21} + x_{22} + x_{23} + x_{24} \\ x_{31} + x_{32} + x_{33} + x_{34} \\ x_{41} + x_{42} + x_{43} + x_{44} \end{bmatrix} = \begin{bmatrix} xK_1 \\ xK_2 \\ xK_3 \\ xK_4 \end{bmatrix}$$

Menotasikan formulasi kedalam persamaan matematika *downsampling* (reduksi) matriks penjumlahan nilai-nilai kolom tiap baris matriks seperti Teorema 1:

$$xK_n = \sum_{l=1}^{m} x_{mn} \qquad (1)$$

Dimana :

xK_n : *Downsampling* (reduksi) baris

x_{mn} : matriks baris kolom

Hasil dari formulasi teorema 10, bisa dibentuk menjadi matriks baru dengan ukuran *m x 1* atau 10x1, seperti berikut :

$$xK_n = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 2 \\ 2 \\ 6 \\ 2 \\ 2 \\ 2 \\ 0 \end{bmatrix}$$

Atau langsung dijadikan vektor *m* kolom atau 10 kolom :

$$vektorxK_n = vxK_n = [xK_1 \dots xK_n]$$

atau

$$vxK_n = [xK_1 \dots xK_n] \qquad (2)$$

Hasil dari matriks 10x10 adalah vektor 10 kolom adalah sebagai berikut:

$$vxK_n = [0\ 1\ 2\ 2\ 2\ 6\ 2\ 2\ 2\ 0\ ]$$

Untuk memperkecil nilai input JST, misalnya dibatasi hanya nilai $-1 \leq xK_n \leq 1$, maka setiap nilai xK_n dari vektor vxK_n dibagi dengan jumlah kolom awal (jK) dalam hal ini $jK = 10$.

$$vxK_n = \left[\frac{xK_1}{jK} \ldots \frac{xK_n}{jK}\right] \tag{3}$$

hasilnya

$$vxK_n = \left[\frac{0}{10}\ \frac{1}{10}\ \frac{2}{10}\ \frac{2}{10}\ \frac{2}{10}\ \frac{6}{10}\ \frac{2}{10}\ \frac{2}{10}\ \frac{2}{10}\ \frac{0}{10}\right]$$

***Downsampling* (Reduksi) Baris (DS-B) Matriks**

Untuk *Downsampling* (Reduksi) Baris (DS-B) Matriks adalah kebalikan dari *downsampling* (reduksi) kolom matriks, dimana semua nilai-nilai dalam baris dijumlahkan dalam setiap kolom atas kebawah sehingga menghasilkan matriks baru *1 x n* atau vektor dengan *n* kolom.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

direduksi menjadi

$$(xB) = [0 \quad 1 \quad 5 \quad 1 \quad 2 \quad 2 \quad 6 \quad 1 \quad 0 \quad 0]$$

Jika melihat matriks citra di atas, hasil *dowsampling* (reduksi) baris dari matriks citra adalah 10x10 untuk baris adalah matriks 1x10 dan jika divektorkan menghasilkan vektor 10 kolom. Kalau diformulasikan secara umum untuk sebuah penjumlahan nilai-nilai baris pada setiap kolom matriks citra diatas sebagai berikut :

$$\begin{array}{rcccccccc} xB_1 & = & x_{11} & + & x_{21} & + & \ldots & + & x_{n1} \\ xB_2 & = & x_{12} & + & x_{22} & + & \ldots & + & x_{n2} \\ \vdots & = & \ldots & + & \ldots & + & \ldots & + & \ldots \\ \vdots & = & \ldots & + & \ldots & + & \ldots & + & \ldots \\ xB_n & = & x_{1m} & + & x_{2m} & + & \ldots & + & x_{nm} \end{array}$$

Untuk menotasikan formulasi kedalam persamaan matematika *downsampling* matriks penjumlahan nilai-nilai baris setiap kolom kolom sebagai berikut (Teorema 4) :

$$xB_n = \sum_{l=1}^{n} x_{nm} \quad (4)$$

Dimana :

xB_n : *Downsampling* (reduksi) kolom

x_{nm} : matriks kolom baris

Hasil dari formulasi teorema 12, bisa dibentuk menjadi matriks baru dengan ukuran *1 x m* atau 1 x 10, seperti berikut :

$$x_{kn} = [0 \quad 1 \quad 5 \quad 1 \quad 2 \quad 2 \quad 6 \quad 1 \quad 0 \quad 0]$$

Atau langsung dijadikan vektor *m* kolom atau 10 kolom:

$$vektorxB_n = vxB_n = [xB_1 \ldots xB_n]$$

Atau

$$vxB_n = [xB_1 \ldots xB_n] \quad (5)$$

Jika merujuk pada matriks citra sebelumnya, maka vektor 10 kolom adalah sebagai berikut :

$$vxB_n = [0\ 1\ 5\ 1\ 2\ 2\ 6\ 1\ 0\ 0]$$

Untuk memperkecil nilai input JST, kisaran nilai antar $-1 \leq xB_n \leq 1$, maka setiap nilai xB_n dari vektor vxB_n dibagi dengan jumlah bsris awal (jB) dalam hal ini $jB = 10$.

$$vxB_n = \left[\frac{xB_1}{jB} \ldots \frac{xB_n}{jB}\right] \quad (6)$$

$$vxB_n = \left[\frac{0}{10}\ \frac{1}{10}\ \frac{5}{10}\ \frac{1}{10}\ \frac{2}{10}\ \frac{2}{10}\ \frac{6}{10}\ \frac{1}{10}\ \frac{0}{10}\ \frac{0}{10}\right]$$

Downsampling (Reduksi) Kolom dan Baris (DS-KB) Matriks

Downsampling (Reduksi) Baris dan Kolom (DS-KB) Matriks adalah menggabungkan hasil perhitungan vxK_n dan vxB_n kedalam vektor baru $vxKB_{kn}$ secara berurutan.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad direduksi(xK) = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 2 \\ 2 \\ 6 \\ 2 \\ 2 \\ 2 \\ 0 \end{bmatrix}$$

$$direduksi$$

$$(xB) = [0 \quad 1 \quad 5 \quad 2 \quad 2 \quad 2 \quad 6 \quad 1 \quad 0 \quad 0]$$

Hasil *downsampling* (reduksi) dari matriks citra 10x10 gabungan vxB_n dan vxK_n menghasilkan vektor $vxKB_{kn}$ yaitu vektor 20:

$$(vxKB_n) = [\underbrace{0\,1\,2\,2\,2\,6\,2\,2\,2\,0}_{reduksi\ baris}\ \underbrace{0\,1\,5\,1\,2\,2\,6\,1\,0\,0}_{reduksi\ kolom}\ ]$$

Untuk gabungan ini diformulasikan dengan persamaan 14.1.

$$vxKB_n = [xK_1 \ldots xK_n \ xB_1 \ldots xB_n] \tag{7}$$

Untuk mengecilkan nilai *input* JST kisaran nilai antar $-1 \leq xK_n \leq 1$ atau $-1 \leq xB_n \leq 1$, maka setiap nilai xK_n dari vektor vxK_n dibagi dengan jumlah baris awal (jK) dalam hal ini $jK = 10$, begitu juga dengan setiap nilai xB_n dari vektor $vxBK_n$ dibagi dengan jumlah kolom awal (jB) dalam hal ini $jB = 10$.

$$vxKB_n = \left[\frac{xK_1}{jK} \ldots \frac{xK_n}{jK} \frac{xB_1}{jB} \ldots \frac{xB_n}{jB}\right] \tag{8}$$

Ketiga *downsampling* yang telah di jelaskan diatas bisa menjadi *input* JST untuk pengenalan huruf ataupun yang lainnya.

Jaringan Saraf Tiruan

Jaringan Saraf Tiruan (JST) merupakan terjemahan dari Artificial Neural Network (ANN), JST adalah merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut (Kusumadewi 2003) untuk memecahkan permasalahan tertentu, misal estimasi, prediksi, klasifikasi, segmentasi (*clustering*), pengenalan pola, dll.

Analogi JST dengan Jaringan Saraf Biologis (JSB), Node/*input* pada JST disebut Badan Sel (Soma) pada JSB, *input* pada JST maka pada JSB disebut

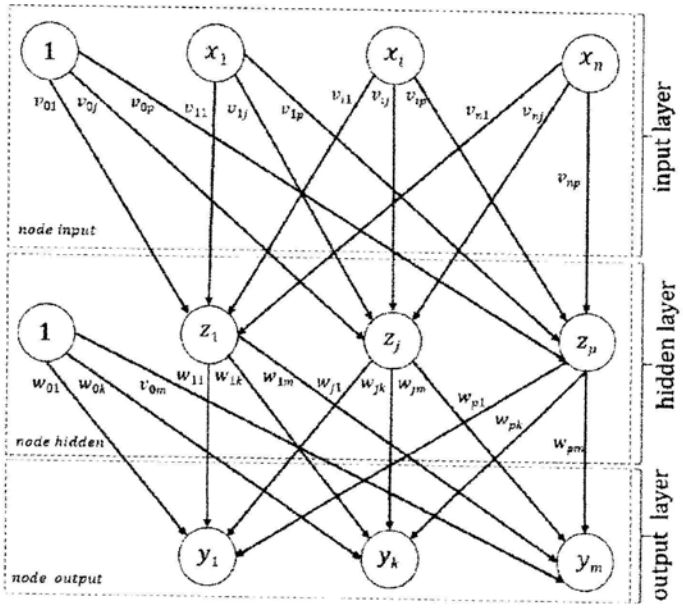
dengan dendrit, *Output* pada JST disebut Akson pada JSB, dan Bobot pada JST disebut dengan Sinapsis pada JSB.

Model pembelajaran JST hadir dalam 2 bentuk:

1. *Supervised learning*, model belajar yang membutuhkan target sebagai acuan arah pembelajaran – supervisi. Dapat diilustrasikan seperti pembelajaran di kelas dimana terdapat seorang Guru yang mengarahkan proses pembelajaran.

2. *Unsupervised learning*, model belajar yang tidak membutuhkan target sebagai acuan arah pembelajaran. Dapat diilustrasikan seperti pembelajaran yang dilakukan sendiri (otodidak).

JST *Backpropagation (BP)*

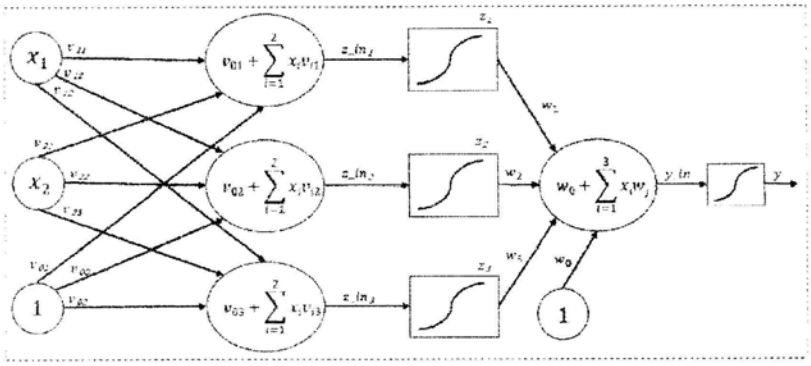


Gambar 2 JST *BP* dengan satu jaringan

Backpropagation (BP) merupakan algoritme pembelajaran yang terawasi dan biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyinya (Kusumadewi 2003). Algoritme *BP* 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. Pada saat perambatan maju, neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi *sigmoid*, yaitu:

$$f(x) = \frac{1}{1 + x^{-x}} \qquad (9)$$

Arsitektur *BP* terdiri dari *input layer*, *hidden layer*, dan *output layer*, perhatikan Gambar 3.



Gambar 3 Arsitektur JST BP fungsi logika XOR input

Algoritme *Backpropagation* (BP)

Ada beberapa langkah yang harus dilalui dalam mengerjakan algoritme BP, mulai dari inisiasi bobot, tahap perambatan maju (*feedforward*), tahap perambatan balik (*backforward*), tahap perubahan bobot dan bias dan dibagian akhir harus melihat kondisi, apakah sudah memenuhi atau tidak? dan selama belum memenuhi pada interval iterasi yang diberikan maka proses akan berulang, adapun langkah-langkah sebagai berikut:

- Tahap Inisiasi dan pengkondisian
  1. Inisiasi bobot, gunakan bobot awal dengan nilai random yang cukup kecil.
  2. Selama kondisi berhenti bernilai *false*, kerjakan pointer-pointer berikutnya.
- Tahap perambatan maju (*feedforward*)
  1. Tiap-tiap unit (*node*) *input* (x_i, $i=1,2,3,\ldots,n$) menerima sinyal x_i dan melanjutkan sinyal tersebut keseluruh unit pada lapisan tersembunyi (*hidden layer*).
  2. Tiap-tiap unit (*node*) tersembunyi (Z_i, $i=1,2,3,\ldots,p$) menjumlahkan bobot sinyal *input* dengan persamaan berikut,

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \tag{10}$$

dan menerapkan fungsi aktifasi untuk menghitung sinyal *output*-nya:

$$z_j = \int (z\_in_j) \tag{11}$$

karena yang digunakan fungsi *sigmoid* maka:

$$z_j = \frac{1}{1 + exp^{(-z\_in_j)}} \tag{12}$$

3. Tiap-tiap unit *output* (y_k $k=1,2,3,\dots,m$) menjumlahkan bobot sinyal *input*.

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_i \, w_{jk} \tag{13}$$

kemudian menerapkan fungsi aktifasi,

$$y_k = \int (y\_in_k) \tag{14}$$

- Tahap perambatan-balik (*backforward*)
  1. Tiap-tiap unit *output* (y_k, k=1,2,3,…,m) menerima pola target yang sesuai dengan pola *input* pelatihan, selanjutnya hitung *error* dengan persamaan berikut,

$$\delta_k = (t_k - y_k) f'(y\_in_k) \tag{15}$$

f' adalah turunan dari fungsi aktifasi, kemudian hitung koreksi bobot dengan persamaan berikut,

$$\Delta w_{jk} = \alpha \, \delta_k \, z_k \tag{16}$$

dan menghitung koreksi bias dengan persamaan berikut,

$$\Delta w_{0k} = \alpha \, \delta_k \tag{17}$$

sekaligus mengirimkan δ_k ke unit-unit yang ada di lapisan paling akhir.

2. Tiap-tiap unit tersembunyi (z_j, $j = 1,2,3,\dots,p$) menjumlahkan delta *input*-nya (dari unit-unit yang berada pada lapisan dibawahnya):

$$\delta\_in_j = \sum_{k=1}^{m} \delta_i \, w_{jk} \tag{18}$$

Untuk menghitung informasi *error*, kalikan nilai δ_in_j dengan turunan dari fungsi aktifasinya,

$$\delta_j = \delta\_in_j \, f'(z\_in_j) \tag{19}$$

Kemudian hitung koreksi bobot dengan persamaan berikut:

$$\Delta v_{jk} = \alpha \, \delta_j \, x_i \tag{20}$$

- Tahap perubahan bobot dan bias
  1. Tiap-tiap unit output (y_k, $k = 1,2,3, \dots, m$) dilakukan perubahan bobot dan bias ($j = 1,2,3, \dots, p$) dengan persamaan berikut:

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (21)$$

  Setiap unit tersembunyi (z_j, $j = 1,2,3, \dots, p$) dilakukan perubahan bobot dan bias ($j = 1,2,3, \dots, n$) dengan persamaan berikut,

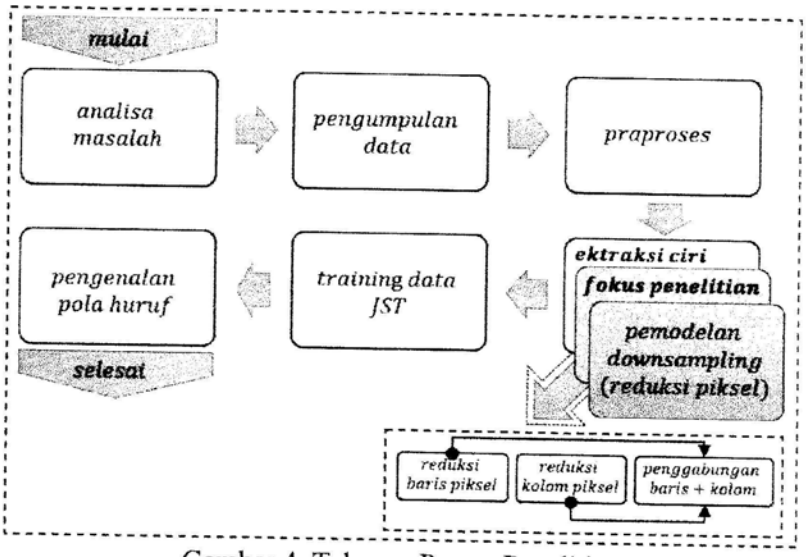
$$v_{ij}(baru) = v_{ik}(l\ ama) + \Delta v_{ik} \quad (22)$$

  2. Tes kondisi berhenti.

3 METODE

Tahapan Penelitian

Tahapan penelitian yang akan dilakukan terdiri dari 6 tahapan, yaitu: Tahap Analisa Masalah, Tahap Penyiapan (Pengumpulan) Data, Tahap Praproses, Tahap Ekstraksi Ciri, Tahap *Training* Data JST, dan Tahap Pengenalan Polah Huruf.



Gambar 4 Tahapan Proses Penelitian

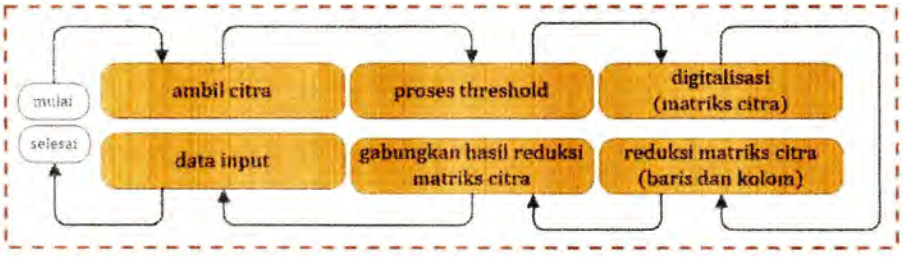
Analisa Masalah

Tahap pertama ialah menganalisa permasalahan yang terjadi saat ini, yaitu masih rendahnya akurasi dan waktu yang dibutuhkan dalam pengenalan pola pemeriksaan hasil ujian. Hasil analisis akan menjadi dasar untuk membangun model *downsampling* untuk ekstraksi ciri untuk pengenalan pola JST.

Penyiapan (Pengumpulan) Data

Sumber Data/Koleksi Data yang akan digunakan pada penelitian ini adalah data yang telah digunakan pada penelitan Dwi *et al* (2011), data tersebut telah melalui proses *cropping*, sehingga pada penelitian ini tidak akan melalui proses tersebut. Formulir sampel tulisan tangan yang sudah terisi sebanyak 105 formulir yang diambil dari orang yang berbeda dan berusia di atas 30 tahun. Setiap orang menuliskan huruf 'A' sampai dengan 'Z', masing-masing sebanyak 10 kali pada formulir isian tersebut, yang digunakan pada penelitian ini hanya huruf 'A' sampai dengan 'E'. Pengambilan sampel dilakukan dengan variasi tempat, pencahayaan, dan situasi, dengan harapan diperoleh lebih banyak variasi bentuk huruf, serta untuk menfasilitasi kondisi lapangan (tempat ujian) yang sesungguhnya. Alat tulis yang digunakan tidak hanya pensil 2B, namun dengan alat tulisan yang lain, seperti bolpoin, spidol, dan lain-lain dengan berbagai warna. Dari penyebaran formulir isian, jumlah sampel huruf yang diperoleh adalah 5 250 karakter.

Praproses



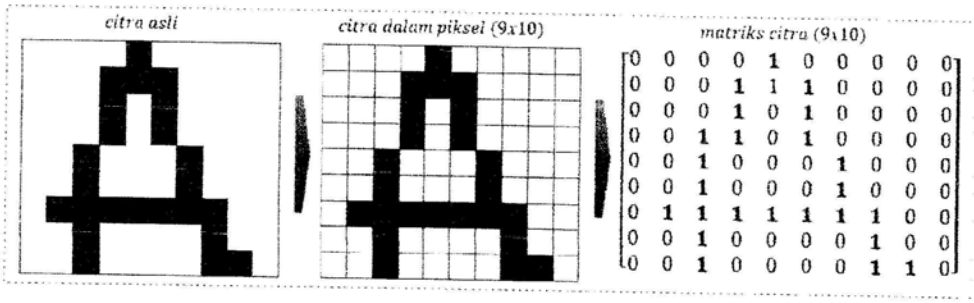
Gambar 5 Alur praproses, *threshold*, digitalisasi dan reduksi matriks citra

Suatu citra tidak serta merta langsung bisa diolah kedalam JST, JST mempersyaratkan bahwa nilai *input* harus berupa nilai numerik, maka diperlukan praproses dimulai dengan menyederhanakan citra digital sehingga siap olah, praproses yang pertama adalah mengetahui besaran piksel dari sebuah citra, pada penelitian ini akan menggunakan piksel yang sama dari setiap gambar, tidak jadi masalah baris dan kolom tidak sama, akan tetapi konsisten pada setiap gambar, misalnya menggunakan citra dengan dimensi 50x50, maka semua gambar harus dengan ukuran yang sama.

Proses berikutnya adalah citra warna diubah menjadi warna keabuan (*grayscale*), dan berlanjut pada proses pengambangan (*thresholding*) citra yaitu mengubah citra yang sudah berderajat keabuan menjadi citra biner atau hitam putih sehingga jelas membedakan mana latar belakang (*background*) cita dan objek citra, skala keabuan dimulai dari 0 sampai dengan 255 yang disebut dengan level. *Threshold* yang digunakan adalah pendekatan metode Otsu yakni melakukan analisis diskriminan yaitu menentukan suatu variabel yang dapat membedakan antara dua atau lebih kelompok yang muncul secara alami. Analisis Diskriminan akan memaksimumkan variabel tersebut agar dapat membagi objek dan latar belakang (*background*). Hasil dari Algoritme Otsu diatas yang menjadi dasar untuk proses numersisasi untuk membuat matriks citra. Untuk mencapai tahap digitalisasi diperlukan beberapa tahap, Pertama, diperlukan karakter atau bagian dari karakter perlu diekstraksi dari representasi bergambar, Kedua, Pemisahan karakter menjadi segmen-segmen(Perwej *et al.* 2011), Ketiga, proses digitalisasi (Gambar 6).

Ekstraksi ciri

Citra yang telah dikonversi dan sudah dapat diidentifikasi dengan tepat mana sebagai *background* citra dan mana sebagai objek citra. proses konversi melibatkan digitalisasi *grid* segmen, karena JST perlu masukan dalam bentuk biner (0 dan 1) (Perwej *et al.* 2011), 0 untuk warna *background* citra (warna putih) dan 1 untuk objek citra (warna hitam), lihat Gambar 10.



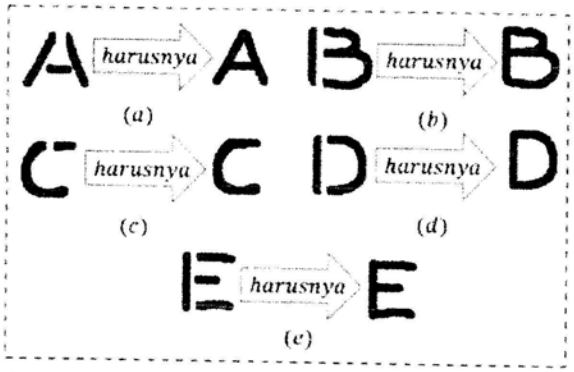
Gambar 6 Proses konversi dari citra menjadi matriks citra

Nilai dari matriks citra yang sudah memenuhi untuk menjadi nilai numerik untuk *input-input* JST, nilai-nilai matriks citra ditampung kedalam *array* dimensi *nxn* untuk kemudian dilakukan reduksi dengan teknik *downsampling* (reduksi). Dalam penelitian ini akan menggunakan *downsampling* (reduksi) kolom dan *downsampling (reduction)* baris, akan dilakukan penggabungan dari kedua cara tersebut.

Analisa awal dengan Reduksi Kolom dan Baris matriks citra

Gaya orang menuliskan sebuah huruf tidak semua sama, mungkin benar dari sisi pemaknaan akan tetapi salah dari sisi tampilan atau kasat mata. Dibawah ini akan diberikan contoh kesalahan-kesalahan dari sisi tampilan dan dijadikan sebagai *input* dalam sebuah pengenalan, sekaligus analisanya, sebagai berikut:

a. Kesalahan tulisan dari sisi kolom dan analisa logis untuk *downsampling* (reduksi) kolom matriks citra, kesalah tulisan yang dilihat atas ke bawah ada ruang atau piksel yang memperantarai (kosong) dari tulisan yang seharusnya tidak ada, contoh (lihat Gambar 11):



Gambar 7 Contoh penulisan huruf yang salah sisi kolom

Yang dijadikan contoh pada Tabel 1 adalah huruf kapital 'E', setelah melalui proses digitalisasi dan binerisasi maka didapatkan matriks citra ($15x15$).

Tabel 1 Perbandingan 2 citra huruf 'E' (salah dan benar)

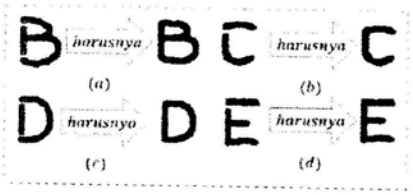
No	Citra	Matriks Citra hasil Reduksi
1		$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&1&1&1&1&1&1&1&1&0&0 \\ 0&0&0&1&1&0&0&0&0&0&1&1&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&1&1&1&1&1&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&1&1&1&1&1&1&1&0&0&0 \\ 0&0&0&1&1&1&1&1&1&1&1&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix} (xK1) = \begin{bmatrix} 0 \\ 0 \\ 10 \\ 4 \\ 2 \\ 2 \\ 7 \\ 7 \\ 2 \\ 2 \\ 2 \\ 9 \\ 9 \\ 0 \\ 0 \end{bmatrix}$
2		$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&1&1&1&1&1&1&1&1&0 \\ 0&0&0&1&1&0&0&0&0&0&0&1&1&1&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&1&1&1&1&1&0&0&0&0 \\ 0&0&0&1&1&0&1&1&1&1&1&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&1&0&1&1&1&1&1&1&1&1&0 \\ 0&0&0&1&1&0&1&1&1&1&1&1&1&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix} (xK2) = \begin{bmatrix} 0 \\ 0 \\ 10 \\ 5 \\ 2 \\ 2 \\ 7 \\ 7 \\ 2 \\ 2 \\ 2 \\ 9 \\ 9 \\ 0 \\ 0 \end{bmatrix}$

Kedua citra huruf kapital 'E' yang dijadikan contoh diatas memiliki kemiripan, akan tetapi juga memiliki sisi perbedaan yang mencolok, citra nomor 1 memiliki ketepatan tulisan yang sangat tinggi dan dari sisi tampilan memenuhi keriteria untuk disebut sebagai huruf kapital 'E', sekarang bandingkan citra nomor 2 yang tidak memiliki ketepatan tulisan, walaupan pada tampilan mengikuti kemiripan dengan huruf kapital 'E', akan tetapi ada beberapa titik yang memisahkan antara garis vertikal disebelah kiri dan 3 garis horizontal disebalah kanan. Ditinjau dari matriks citra, antara matriks citra nomor 1 dan matriks citra nomor 2 maka titik nilai matriks citra yang berbeda berada pada titik berikut :

$$(x_{3,6}, x_{3,14}, x_{4,11}, x_{4,13}, x_{4,14}, x_{7,6}, x_{7,11}, x_{8,6}, x_{8,11}, x_{12,6}, x_{12,14}, x_{13,6}, x_{13,14})$$

tercatat ada 13 titik perbedaan, sedangkan jika dibandingkan dengan hasil reduksi yang ditunjukkan $xK1$ dan $xK2$ ada pada titik $x_{4,1}$ dengan nilai 4 dan 5.

b. Kesalahan tulisan dari sisi baris dan analisa logis untuk *downsampling* baris matriks citra, kesalahan tulisan yang dilihat dari adanya ruang atau piksel yang memperantarai dari tulisan yang seharusnya tidak ada yang dilihat dari kiri ke kanan (baris), contoh (lihat Gambar 12) :



Gambar 8 Contoh penulisan huruf yang salah sisi baris

Huruf kapital 'E' akan dijadikan sebagai contoh pada Tabel 2, setelah melalui proses binerisasi maka dihasilkan matriks citra $(15x15)$ seperti pada Tabel 2 berikut:

Tabel 2 Perbandingan 2 citra huruf 'E' (salah dan benar)

No	Citra	Matriks Citra hasil Reduksi
1	E	$(xB1) = [0\ 0\ 0\ 10\ 11\ 6\ 6\ 6\ 6\ 6\ 4\ 4\ 0\ 0\ 0]$
2	E	$(xB2) = [0\ 0\ 0\ 11\ 11\ 6\ 6\ 6\ 6\ 6\ 4\ 4\ 0\ 0\ 0]$

Gambar pada nomor 1 memperlihatkan sisi kesempurnaan dalam penulisan huruf 'E', akan tetapi untuk citra nomor 2, walau kelihatan sama dengan citra nomor 1, citra nomor 2 menunjukkan keterpisahan dari garis vertikal dan garis horizontal untuk bagian atas, titik celah tersebutlah menjadi pembeda di antara keduanya sekaligus menjadi tumpuan pengamatan. Jika diamati dari sisi perbedaan titik matriks citra nomor 1 dan matriks citra nomor 2 maka inilah titik-titik yang berbeda sebagai berikut:

$$\begin{pmatrix} x_{2.4}, x_{2.5}, x_{2.6}, x_{2.7}, x_{2.8}, x_{2.9}, x_{2.10}, x_{2.11}, x_{2.12}, \\ x_{3.4}, x_{4.4}, x_{4.5}, x_{4.6}, x_{4.7}, x_{4.8}, x_{4.9}, x_{4.10}, x_{4.11}, x_{4.12} \end{pmatrix}$$

tercatat ada 19 titik perbedaan, sedangkan jika dibandingkan dengan hasil reduksi yang ditunjukkan $xB1$ dan $xB2$ ada pada titik $(x_{1.14})$ dengan nilai 10 dan 11.

c. Kesalahan tulisan yang terdiri dari keduanya (sisi baris dan sisi kolom), kesalahan tulisan yang dilihat dari adanya ruang atau piksel yang memperantarai dari tulisan yang seharusnya tidak ada yang dilihat dari kiri ke kanan dan dari atas ke bawah.

Gambar 9 Huruf kapital 'E' yang salah sisi kolom dan baris

Dari hasil pengamatan dari dua bentuk kesalahan tulisan huruf (poin a dan b mewakili c) dan melihat titik perbedaan nilai-nilai matriks citra dari keduanya, begitu juga mengamati hasil reduksi dengan *downsampling* nilai-nilai matriks citra dari kedua citra, dalam pengamatan awal ini disimpulkan bahwa sistem *downsampling* bisa memperkecil tingkat perbedaan dan mempertajam kesamaan nilai *input* dari kedua citra yanga ada, terbukti bahwa telah di-*downsampling* terjadi titik pengurangan perbedaan seperti pada Tabel 3 berikut :

Tabel 3 Tingkat perbedaan *downsampling* dan tanpa *downsampling*

No	Keterangan	Jumlah Titik Perbedaan dan Input			
		Tanpa Downsampling		Downsampling	
		Titik Beda	Input	Titik Beda	Input
1	Kolom	13	225	1	15
2	Baris	19	225	1	15

Untuk besar kecilnya dimensi citra telah dilakukan pengamatan pada dimensi citra $5x5, 7x7, 10x10, 15x15, 25x25$ dan $50x50$, ini dilakukan untuk melihat ukuran dimensi berapakah yang layak efektif untuk dilakukan proses *downsampling* (reduksi)? ternyata semakin kecil dimensi matriks citra maka semakin menurunkan tingkat kelayakan untuk dijadikan sampel, karena nilai-nilai matriks citra semakin menyerupai nilai-nilai matriks yang lain walau berbeda pola. Ambang batas terkecil yang efektif untuk dijadikan sampel adalah matriks citra ukuran $10x10$.

Tabel 4 Hasil pengamatan dimensi citra 5x5 dan 7x7

No	Dimensi Citra	Matriks Citra		Dawnsampling Citra	
		B	E	B	E
1	5x5	00000 01110 01110 11101 01110	00000 01110 01100 01110 00111	0 3 3 3 3	0 3 2 3 3
2	7x7	0000000 0111100 0100010 0001110 0100010 0100010 0111100	0000000 0111110 0111000 0011100 0100010 0001110 0011000	0 4 2 3 2 2 3	0 4 2 3 2 3 2

Hasil pengamatan ini, jelas hanya sebagai hitungan kasar untuk melihat sisi kemiripan data *input* yang ada, masih perlu dibuktikan dengan pelatihan di JST.

Inisiasi penyesuaian variabel matriks citra

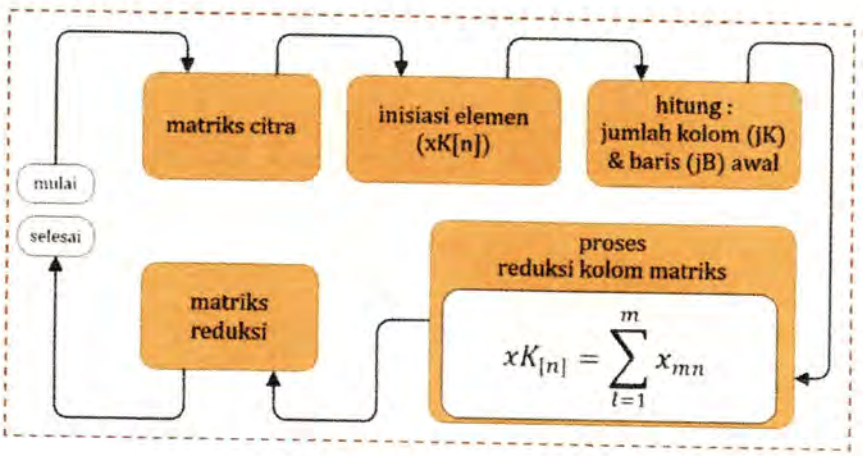
Inisiasi pada penyesuaian variabel matriks citra sebagai berikut:

1. Inisiasi *array* untuk matriks dua dimensi untuk baris ($matriks\_citra[m][n]$).
2. Inisiasi dan menghitung jumlah awal kolom matriks citra (jK).
3. Inisiasi dan menghitung jumlah awal baris matriks citra (jB).

Downsampling (Reduksi) Matriks Citra

Downsampling (reduksi) matrik citra (DS), akan dibagi menjadi 3 metode, yaitu; Downsampling (Reduksi) Kolom Matriks Citra (DS-K), Downsampling (Reduksi) Baris Matriks Citra (DS-B), dan Downsampling (Reduksi) Kolom dan Baris Matriks Citra (DS-KB).

Downsampling (Reduksi) Kolom Matriks Citra (DS-K)



Gambar 10 Proses Reduksi Kolom Matriks

Mereduksi kolom matriks citra dengan cara menjumlahkan nilai-nilai kolom matriks pada setiap baris matriks dan hasilnya disimpan kedalam *array* satu dimensi, dengan langkah-langkah berikut:

1. Inisiasi *array* dua dimensi yang satu dimensi sebagai penampung hasil reduksi kolom matriks citra($xK[m]$).
2. Reduksi matriks citra dengan perulangan dengan bentuk potongan *pseudocode*.

```
for m ← 0 sampai jk-1
 begin
  x ← 0;
   for n ← 0 sampai jb-1
    begin
      x ← x + matriks_citra[m][n];
```

```
    end;
  xK[m] ← x;
end;
```

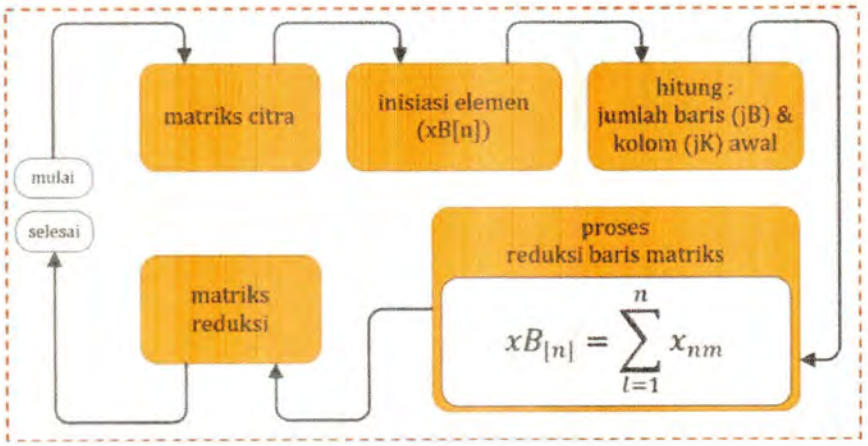
3. Sampai pada nilai hasil reduksi tertampung kedalam variabel *array* (xK).

$$\begin{bmatrix} x_{11} & + & x_{12} & + & \dots & + & \dots & + & x_{1n} \\ x_{21} & + & x_{22} & + & \dots & + & \dots & + & x_{2n} \\ \dots & + & \dots & + & \dots & \dots & \dots & \dots & \dots \\ \dots & + & \dots & + & \dots & \dots & \dots & \dots & \dots \\ \dots & + & \dots & + & \dots & \dots & \dots & \dots & \dots \\ x_{m1} & + & x_{m2} & + & \dots & + & \dots & + & x_{mn} \end{bmatrix} \begin{matrix} = xK_1 \\ = xK_2 \\ = \dots \\ = \dots \\ = \dots \\ = xK_n \end{matrix}$$

Sesuai dengan persamaan 1:

$$xK_n = \sum_{l=1}^{m} x_{mn}$$

Downsampling (Reduksi) Baris Matriks Citra (DS-B)



Gambar 11 Proses Reduksi Baris Matriks

Mereduksi kolom matriks citra dengan menjumlahkan nilai-nilai tiap baris yang ada dikolom matriks tersebut dan hasilnya disimpan kedalam *array* satu dimensi, dengan langkah-langkah berikut:

1. Inisiasi *array* satu dimensi sebagai penampung hasil reduksi baris matriks citra ($xk[m]$).
2. Perulangan untuk reduksi matriks citra (*pseudocode*).

```
for m ← 0 sampai jb-1
 begin
  x ← 0;
   for n ← 0 sampai jK-1
    begin
      x ← x + matriks_citra[m][n];
```

```
    end;
  xB[m] ← x;
end;
```

3. Sampai pada nilai hasil reduksi tertampung kedalam variabel *array* (xB).

$$\begin{bmatrix} x_{11} & + & x_{12} & + & \dots & + & \dots & + & x_{1n} \\ x_{21} & + & x_{22} & + & \dots & + & \dots & + & x_{2n} \\ \dots & + & \dots & + & \dots & \dots & \dots & \dots & \dots \\ \dots & + & \dots & + & \dots & \dots & \dots & \dots & \dots \\ \dots & + & \dots & + & \dots & \dots & \dots & \dots & \dots \\ x_{m1} & + & x_{m2} & + & \dots & + & \dots & + & x_{mn} \end{bmatrix}$$

$$= \begin{bmatrix} xB_1 & \dots & \dots & \dots & xB_n \end{bmatrix}$$

Sesuai dengan Persamaan 2 :

$$xB_n = \sum_{l=1}^{n} x_{nm}$$

Downsampling (Reduksi) Kolom dan Baris Matriks Citra (DS-KB)

Tahap berikutnya dari ektraksi ciri adalah melakukan penggabungan nilai hasil reduksi matriks citra antara baris dan kolom langkah-langkah berikut:

1. Inisiasi *array* dua dimensi yang nantinya akan disebut sebagai vektor n kolom ($vxKBn$).
2. Set vektor $vxKBn$ dengan $setlength(vxKBn, jK + jB)$, code dalam bentuk *pascal*.
3. Perulangan untuk menyatukan hasil reduksi matriks citra baris dan kolom (*pseudocode*).

```
x ← 1;
//isi vektor dengan hasil reduksi kolom matriks
for i ← 0 sampai jk-1
 begin
  x ← x + 1;
    begin
      vxBKn[x] ← xK[i]/jK;
    end;
 end;

//isi vektor dengan hasil reduksi baris matriks
for i ← 0 sampai jb-1
 begin
  x ← x + 1;
    begin
      vxBKn[x] ← xB[i]/jB;
    end;
 end;
```

4. Nilai yang tertampung dalam variabel *array* $vxBKn$ adalah sebuah variabel yang sudah siap menjadi *input* JST. Sesuai dengan Persamaam 8:

$$vxKB_n = \left[\frac{xK_1}{jK} \ldots \frac{xK_n}{jK} \frac{xB_1}{jB} \ldots \frac{xB_n}{jB}\right]$$

Training Data pada JST

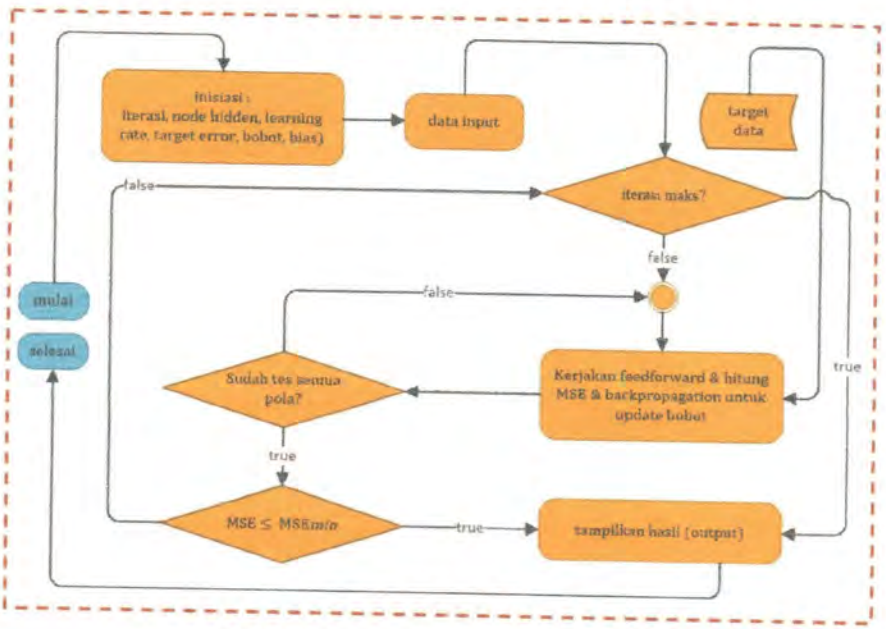
Setelah melalui tahap normalisasi sehingga citra digital berhasil dikonversi menjadi sebuah data numerik dan melakukan *downsampling* dan terakhir dari ekstrak ciri adalah membentuk vektor sehingga menjadi nilai-nilai *input* JST BP dan kemudian memulai proses *training* data. Pada proses JST-BP akan melalui 3 lapisan (*layer*), yaitu; *input layer*, *hidden layer*, dan *output layer*. Untuk *input layer* akan mengikuti hasil algoritme DS, untuk DS-K 50 *input*, DS-B 50 *input* dan DS-KB 100 *input*. Sementara *output layer* memberikan 5 *output*. Yang menjadi bervariasi adalah *hidden* layer.

Jumlah *hidden layer* optimal cukup satu saja (Fausett 1994), bertambahnya jumlah *hidden layer* berefek pada penurunan akurasi (Hagan, *et al.* 2002), dalam percobaan arsitektur JST yang dilakukan mencoba mulai dari jumlah *hidden layer* 2, 3, 4, dan 5, akurasi paling rendah ditunjukkan pada jumlah *hidden layer* 5. Berbeda dengan pemilihan jumlah neuron pada *hidden layer*, memutuskan jumlah neuron di *hidden layer* adalah bagian yang sangat penting dari arsitektur JST secara keseluruhan, meskipun pada *layer* ini tidak secara langsung berinteraksi dengan lingkungan eksternal, akan tetapi memiliki pengaruh yang sangat besar pada hasil akhir (Panchal *et al.* 2011).

Menggunakan terlalu sedikit neuron pada *hidden layer* dapat mengakibatkan *underfitting*, dan terlalu sedikit neuron pada *hidden layer* dapat mengakibatkan *overfitting* (Sharma *et al.* 2013). *Underfitting* akan terjadi ketika serangkaian *input* banyak dan jumlah neuron *hidden layer* sedikit, dan *Overfitting* akan terjadi ketika *input* sangat sedikit dan jumlah neuron pada *hidden layer* terlalu sedikit, banyaknya jumlah neuron pada *hidden layer* akan meningkatkan waktu pada saat *training* data dan *testing data*.

Tiga kemungkinan permasalahan diatas, akhirnya memaksa kita untuk melakukan pendekatan *trial and error* (mencoba dan kemudian melihat tingkat error dan akurasinya), hal ini diperkuat oleh Shih (1994) dan Sarangi (2013) tidak ada aturan standar atau nyata mengenai penentuan jumlah neuron pada *hidden layer*, kecuali aturan praktis, umumnya yang terbaik adalah dengan bentuk piramida. Pada penelitian ini akan menggabungkan sistem piramida yang sejalan dengan dua dari tiga aturan metode *rule-of-thumb* untuk jumlah neuron pada *hidden layer*, yaitu; Jumlah neuron *hidden layer* 2/3 (atau 70% sampai 90%) dari ukuran *input layer* (Karsoliya 2012) dan dapat dilakukan pengurangan dan penambahan jika dibutuhkan dikemudian hari (Boger *et al.* 1997), dan jumlah neuron *hidden layer* ialah antara jumlah *input layer* dan *output layer* (Panchal *et al.* 2011).

Peningkatan jumlah neuron lebih lanjut akan meningkatkan kompleksitas JST serta menambah waktu pelatihan, pemakaian nilai parameter jumlah neuron pada *hidden layer* tepat sangat berperan dalam meningkatkan efisiensi berbagai aplikasi jaringan saraf (Jadhav *et al.* 2016), skenario/desain ujicoba yang dilakukan Jadhav akan menjadi acuan ujicoba pada *training* data dengan beberapa variasi jumlah neuron.



Gambar 12 Proses Training Data di JST

Pengenalan Pola Huruf

Tahap terakhir metode ini adalah mengenali pola huruf, data yang sudah melaui proses *training* akan dilihat hasil pengelompokannya melalui pengenalan pola.

Analisa Hasil Penelitian

Hasil yang dicapai pada penelitian ini, akan dianalisa dengan tiga komponen, yaitu; Akurasi, Rentang, dan Waktu. Akurasi adalah prosentase atau hasil rata-rata yang dihasilkan dari pengenalan pola. Rentang (*Range*) adalah selisih antara data dengan nilai terbesar (x_{max}) dengan data nilai yang terkecil (x_{min}) atau dalam formulasi ilmu statistik dikenal dengan $R = x_{max} - x_{min}$, x_n adalah nilai presisi setiap pola yang dalam penelitian ada 5 (x_1, x_2, x_3, x_4, x_5). sedangkan Waktu adalah waktu yang digunakan dalam sekali *training* data atau *testing* data.

Hasil yang akan dianalisa meliput 2 bagian, yaitu; hasil *training* data dan hasil *testing* data.

Alat

Pada penelitian ini adapun alat-alat yang akan digunakan sebagai pembuatan model dan mesin komputer yang digunakan untuk ujicoba adalah seperti pada Tabel 5.

Tabel 5 Spesifikasi Laptop 1 untuk ujicoba

Jenis	Laptop
Merek	Toshiba (Satellite P55W-C)
Prosessor	Intel® Core™ i7-6500U CPU@ 2.500GHz (4 CPUs)
Layar	15.6 inc (resolusi 1920 x 1080)
Memory	8 Gb
Hardisk	1 TB
Windows	Windows 10 Home 64-bit (10.0, Build 10586)

4 HASIL DAN PEMBAHASAN

Penyiapan (Pengumpulan) Data

Perlu diketahui lebih awal bahwa data yang digunakan pada penelitian yang dilakukan adalah menggunakan data penelitian rujukan. Pengumpulan data sampel tulisan melalui formulir yang terisi adalah sebanyak 105 formulir, pemilihan orang dilakukan secara acak dengan jumlah pengisi 105 orang dengan menuliskan huruf 'A' sampai dengan 'Z', dengan kriteria usia 30 tahun keatas sesuai dengan karakteristik mahasiswa UT, dan pengisian formulir dilakukan pada kondisi seadanya yang mewakili keadaan dan kondisi lapangan mahasiswa UT, dengan harapan kriteria ini mewakili karakteristik mahasiswa UT dengan keterbatasan kondisi di lapangan.

Pola yang digunakan pada penelitian ini hanya dari pola huruf 'A' sampai dengan pola huruf 'E', sehingga hanya diperoleh sebanyak 5 250 karakter sampel. Jumlah sampel yang tersebut di atas tidak semua data *training* dan data *testing*, ada proses eliminasi sampel yang penyebabnya karena ada beberapa data yang tidak konsisten.

Data Sampel Tidak Konsisten

Formulir isian yang disebarkan bukanlah dalam bentuk LJU sebenarnya, demikian juga pada saat pengisian bukan situasi ujian nyata, maka analisa yang dapat diberikan bahwa disadari atau tidak disadari pada saat pengisian formulir oleh sebanyak orang yang mengisi formulir, dihadapkan pada kemungkinan kondisi (psikologi) berikut:

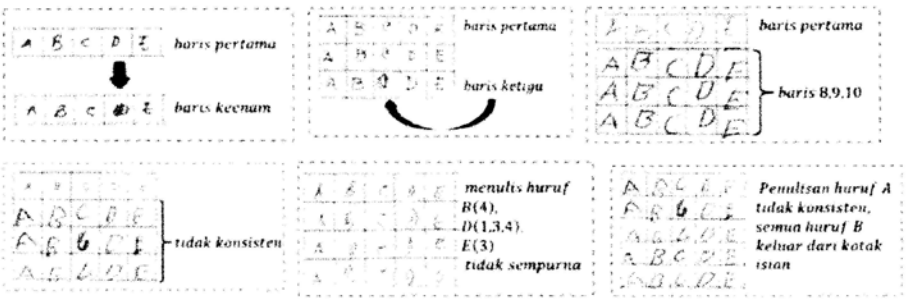
1. Sangat Konsentrasi.
2. Konsentrasi.
3. Kurang Konsentrasi.
4. Sangat Tidak Konsentrasi

Dengan faktor psikologi poin 3 dan 4 di atas sehingga pada saat mengisi formulir kemungkinan berikut bisa terjadi, yaitu;

1. Mengisi apa adanya.
2. Terburu-buru dalam menuliskan huruf.
3. Kurang memperhatikan kotak yang mau diisi sehingga tulisan melewati kotak isian.
4. Salah mengisi huruf pada kotak huruf sehingga terjadi penimpaan tulisan awal.
5. Penulisan pada baris pertama bagus akan tetapi pada baris berikutnya mengalami perubahan.

Eiminasi Sampel

Disebabkan adanya beberapa sampel yang tidak konsisten, maka data-data yang tidak konsisten tersebut tidak akan diikutkan (dieliminasi) dalam ujicoba model yang akan dibuat.



Gambar 13 Pengisian formulir diduga telah mengalami *modification* dan tidak konsisten

Pada Gambar 13 adalah contoh-contoh tulisan tangan yang kurang konsisten dan menjadi target eliminasi.

Jumlah Data Sampel setelah proses Eliminasi

Setelah mengalami proses eliminasi, maka data yang didapatkan jumlah data citra tulisan tangan sebelum proses eliminasi 5 250 sampel citra, setelah dilakukan proses eliminasi terdapat 3 820 sampel citra huruf. Sejumlah data yang lolos dari eliminasi tersebut akan data *training* sebanyak 3 210 sampel dan data *testing* sebanyak 610 sampel.

Praproses

Data sampel citra yang telah melalui proses eliminasi kemudian dilakukan proses digitasi, yaitu proses konversi dari image menjadi nilai numerik (digitasi) dengan menggunakan metode Otsu. Nilai-nilai RGB (*Red, Green, Blue*) lebih besar dari nilai *threshold* nilai matriks citranya diberi nilai 0, dan jika nilai RGB lebih kecil sama dengan nilai *threshold* nilai matriks citranya diberi nilai 1.

Ekstraksi Ciri

Hasil dari praproses adalah sebuah matriks citra dengan ukuran 50x50, matriks citra yang dihasilkan kemudian kita terapkan pemodelan *downsampling* (reduksi) piksel ekstraksi ciri. Pada pemodelan *downsampling* dilakukan dengan 3 proses, terdiri dari 2 proses reduksi matriks citra dan 1 proses penggabungan 2 proses reduksi, 2 proses reduksi matriks citra yaitu: reduksi kolom matriks citra dan reduksi baris matriks, dan 1 proses penggabungan adalah proses penggabungan hasil reduksi kolom maktris citra dan hasil reduksi baris matriks citra.

Training Data pada JST

Pada penelitian ini menggunakan arsitektur JST untuk *training* data dengan parameter maksimun iterasi (*epoch*) = 4000, *learning rate* atau laju pembelajaran (α) = 0.5, vektor = 100, Target *error* (MSE) = 0.5, *ouput (class)* = 5, karena pada *training* data kita menggunakan JST-*Backpropagation output* kelas yang ditargetkan adalah [1 0 0 0 0] untuk huruf 'A', [0 1 0 0 0] untuk huruf 'B', [0 0 1 0 0] untuk huruf 'C', [0 0 0 1 0] untuk huruf 'D' dan [0 0 0 0 1] untuk huruf 'E'.

Perancangan ujicoba yang dilakukan adalah mengikut pola Jadhav (2016) dengan mengujicoba algoritme DS pada JST-BP, dengan memberikan desain variasi jumlah neuron pada *hidden layer*, yaitu; 15, 20, 25, 30, 35, dan 40 untuk JST-BP DS-K, DS-B, dan DS-KB, dan berdasarkan iterasi dengan variasi iterasi 500, 1 000, 1 500, 2 000, 2 500, 3 000, 3 5000, 4 000.

Pengenalan Pola Huruf

Seperti dijelaskan sebelumnya bahwa *output* kelas ada lima, 3 210 sampel data, setiap sampel pada akhir iterasi akan dikelompokkan berdasarkan nilai yang dihasilkan pada nilai y(n), 5 *output* yang dihasilkan dalam setiap citra akan memberikan informasi kemana sebuah citra akan mengelompokkan diri, misalnya sebuah citra huruf 'A' menghasilkan *output* target = [*0.1* 0.01 0.005 -0.1 -0.01] maka dia dikelompokkan kedalam huruf 'A' karena yang terbesar adalah 0.1, akan tetapi jika sebuah cita huruf 'B' menghasilkan *output* target = [*0.1* 0.01 0.005 -0.1 -0.01], maka dia akan dikelompokkan ke huruf 'A' walaupun sudah diketahui adalah huruf 'B'.

Hasil Pengujian Model Downsampling (Reduksi) Matriks

Untuk memeriksa fungsi dari metode yang kami usulkan dengan melakukan ujicoba data citra tulisan tangan yang diperoleh dari penelitian rujukan. Selanjutnya, analisis perbandingan yang akan dilakukan untuk mengevaluasi kinerja metode yang diusulkan yaitu dengan mengoptimasi nilai input JST dengan dengan melakukan *downsampling* (reduksi) baris dan kolom matriks citra dengan melakukan pelatihan JST *Backpropagation*.

Jumlah sampel data *training* dan validasi yang akan digunakan setelah melalui proses eliminasi sampel data sebanyak 3 210 karakter huruf tulisan tangan terdiri dari karakter 'A', 'B', 'C', 'D', dan 'E'.

Pada hasil analisa penulis membagi menjadi 7 sub bab, yaitu; Hasil *Downsampling* (Reduksi) Kolom (DS-K) Matriks, Hasil *Downsampling* (Reduksi) Baris (DS-B) Matriks, Hasil Gabungan *Downsampling* (Reduksi) Kolom dan Baris (DS-KB) Matriks, Kenaikan Akurasi Pola Huruf 'B' dan 'C' pada DS-KB, Perbandingan hasil DS-K, DS-B, dan DS-KB dan memilih Teroptimasi, Memilih Jumlah Neuron *Hidden Layer* sebagai Teroptimasi pada DS-KB, dan Hasil *Downsampling* (DS) dengan variasi iterasi.

Hasil Downsampling (Reduksi) Kolom (DS-K) Matriks

Hasil yang ditampilkan dari rangkaian percobaan untuk DS-K matriks pada JST-BP dapat dilihat pada Tabel 10 untuk *training* data berikut:

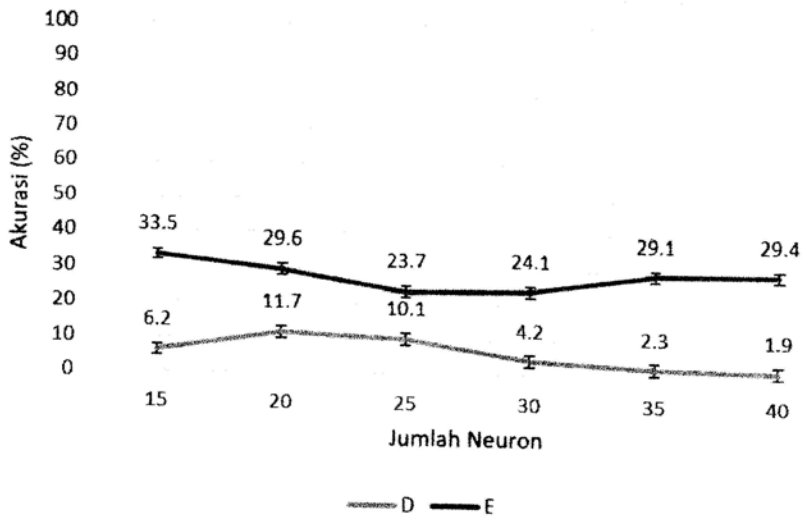
Tabel 6 Analisis Perbandingan JST DS-K pada 5 Pola Karakter pada sejumlah Variasi Neuron untuk *Training* Data

Jumlah neuron *Hidden Layer*	A	B	C	D	E	Akurasi (Rata-rata)	Rentang	Waktu	RMSE
	%	%	%	%	%	%	%	Menit.Detik	
15	67.3	60.1	96.6	93.5	93.5	82.2	36.4	3.19	0.28
20	83.8	58.7	94.9	98.3	95.3	86.2	39.6	4.24	0.23
25	82.4	66.0	95.0	97.5	96.9	87.6	31.5	5.21	0.22
30	90.3	71.7	96.1	98.6	95.6	90.5	26.9	6.23	0.21
35	85.6	68.5	96.8	97.1	97.5	89.1	29.0	6.54	0.11
40	94.4	68.7	92.4	99.2	99.4	90.8	30.7	7.32	0.09

Keterangan : Rentang = $x_{max} - x_{min}$, x_{max} adalah nilai data kinerja tertinggi pada lima pola huruf

Data hasil *training* data pada Tabel 6 dengan menampilkan hasil detil pengenalan setiap huruf. Jumlah neuron pada *hidden layer* dengan kinerja tertinggi ditunjukkan pada jumlah neuron 40 dengan akurasi 99.4% untuk pengenalan huruf 'E', sedangkan untuk pengenalan terendah pada jumlah neuron 5 pada huruf 'B' dengan akurasi 60.1%. Untuk akurasi rata-rata tertinggi terdapat pada jumlah neuron 40 dengan presisi 90.8% dengan RMSE=0.09 dan waktu 7 menit dan 32 detik, walaupun secara rata-rata tinggi pada jumlah neuron 40, tapi tidak jauh berbeda dengan akurasi hasil pada jumlah neuron 30 dengan 90.5% hanya selisih 0.3%, dan nilai ini tidak menjadi otomatis menjadi jumlah neuron dengan kategori efisien, kenapa? Ini disebabkan oleh adanya nilai rentang tinggi, pengenalan pola tertinggi dengan pengenalan terendah dengan selisih tinggi, rentang terendah saja masih pada angka 26.9% pada jumlah neuron 30, jika dilihat dari hasil maka pola huruf 'B' rata-rata mempunyai nilai akurasi yang kecil dibanding dengan 4 pola lainnya dengan selisih yang cukup jauh.

Melihat kinerja pada pengenalan pola huruf 'B' kurang, maka dilakukan observasi lebih dalam dan ternyata sisa pola huruf 'B' yang tidak dikenali sebagai huruf 'B' lebih dikenali sebagai huruf 'D' dan huruf 'E', adapun terhadap huruf 'A' tidak ada dan huruf 'C' terhitung sangat kecil, bisa dilihat pada grafik Gambar 14.



Gambar 14 Prosentase pola huruf 'B' terkenali ke huruf 'D' dan 'E' pada DS-K *training* data

Hasil yang ditunjukkan pada grafik adalah hanya menampilkan prosentase huruf 'B' dikenali sebagai huruf 'D' dan 'E'. hasil di atas memberikan informasi bahwa pola huruf 'B' jika tidak dikenali sebagai dirinya sendiri maka dia lebih cendrung dikenali sebagai huruf yang mirip-mirip dengan polah huruf 'B' yaitu 'D' dan 'E', bahkan lebih banyak ke huruf 'E'.

Sementara dari sisi waktu, bertambahnya jumlah neuron berefek kepada banyaknya waktu yang dibutuhkan, namun hasil menunjukkan dalam sekali *training* data dalam setiap jumlah neuron memtuhkan waktu yang cendrung yang tidak terlalu lama.

Tabel 7 Analisis Perbandingan JST DS-K pada 5 Pola Karakter pada sejumlah Variasi Neuron untuk *Testing* Data

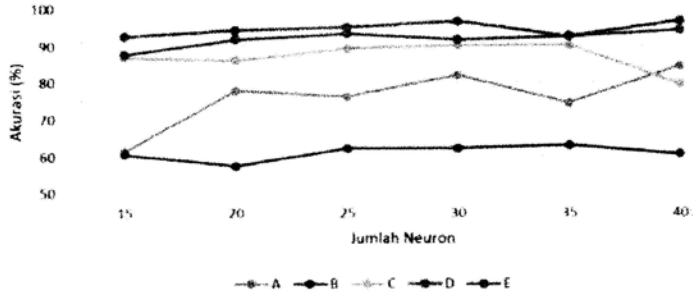
Jumlah neuron *Hidden Layer*	A	B	C	D	E	Akurasi (Rata-rata)	Rentang	Waktu
	%	%	%	%	%	%	%	Menit.Detik
15	61.5	60.7	86.9	92.6	87.7	77.9	32.0	0.06
20	77.9	57.4	86.1	94.3	91.8	81.5	36.9	0.07
25	76.2	62.3	89.3	95.1	93.4	83.3	32.8	0.09
30	82.0	62.3	90.2	96.7	91.8	84.6	34.4	0.11
35	74.4	63.1	90.2	92.6	92.6	82.6	29.5	0.12
40	84.4	60.7	79.5	96.7	94.3	83.1	36.1	0.12

Keterangan : Rentang = $x_{max} - x_{min}$, x_{max} adalah nilai data kinerja tertinggi pada lima pola huruf

Hasil yang ditunjukkan pada Tabel 7 adalah penyajian hasil *testing* data dari JST-BP DS-K, hasil menunjukkan bahwa kinerja terbaik ditunjukkan pada jumlah neuron 30 dan 40 pada pengenalan pola huruf 'D' dengan akurasi yang sama yaitu 96.7%, dan kinerja terendah terdapat pada jumlah neuron 5 huruf 'B' dengan

akurasi 60.7%. Dilihat dari rata-rata maka kinerja terbaik ditunjukkan pada desain jumlah neuron 30 dengan akurasi 84.6%, namun perlu diketahui bahwa pengenalan pada setiap huruf tidak merata. Pengenalan pada huruf 'B' mengikuti pola *training* data, nilai yang dihasilkan pada setiap desain jumlah neuron hanya berada pada akurasi 63.1% kebawah. Untuk rentang didapatkan selisih yang tinggi, nilai terendah masih pada angka 29.5% pada jumlah neuron 35. Dan waktu *testing* data tidak membutuhkan waktu yang lama.

Antara hasil *training* data dan *testing* data, pada dasarnya simulasi yang ditampilkan hampir sama, dari sisi rata-rata, desain jumlah neuron 30 didaulat sebagai berkinerja bagus, akan tetapi hal ini tidak efisien dengan tidak terjadinya nilai rentang tinggi (tidak terjadi pemertaan pengenalan antar pola). Pola huruf 'B' yang menjadi terendah dari hasil *training* data dan *testing* data, dan sisa yang tidak dikenali juga sama mengarah ke huruf yang mirip dengan pola huruf 'B' yaitu pola huruf 'D' dan pola huruf 'E', maka dengan begitu persoalan utama dari DS-K adalah kurang mampu mengenali pola huruf 'B' baik *training* data maupun *testing* data.



Gambar 15 Hasil perbandingan pengenalan pola antar huruf *training* data DS-K

Hasil Downsampling (Reduksi) Baris (DS-B) Matriks

Hasil yang ditampilkan dari rangkaian percobaan untuk DS-B matriks pada JST-BP dapat dilihat pada Tabel 8 untuk *training* data berikut:

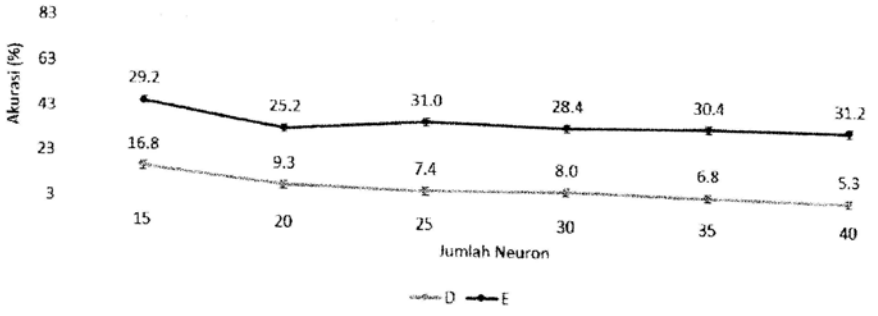
Tabel 8 Analisis Perbandingan JST DS-B pada 5 Pola Karakter pada sejumlah Variasi Neuron untuk *Training* Data

Jumlah neuron *Hidden Layer*	A	B	C	D	E	Akurasi (Rata-rata)	Rentang	Waktu	RMSE
	%	%	%	%	%	%	%	Menit.Detik	
15	73.8	65.4	54.0	76.0	90.4	71.9	36.4	3.21	0.34
20	60.0	80.2	65.5	80.2	87.4	74.7	27.4	4.32	0.31
25	63.2	72.7	61.6	82.7	93.8	74.8	32.2	5.36	0.30
30	70.9	77.5	63.6	88.0	91.1	78.2	27.6	6.14	0.28
35	71.5	76.5	62.8	88.2	90.2	77.8	27.4	6.42	0.27
40	70.2	78.6	63.5	88.3	91.3	78.4	27.8	7.44	0.26

Keterangan : Rentang = $X_{max} - X_{min}$, X_{max} adalah nilai data kinerja tertinggi pada lima pola huruf

Data hasil *training* data pada Tabel 8 dengan menampilkan hasil detil pengenalan setiap huruf. Jumlah neuron pada *hidden layer* dengan kinerja tertinggi ditunjukkan pada jumlah neuron 25 dengan akurasi 93.8% untuk pengenalan huruf 'E', sedangkan untuk pengenalan terendah pada jumlah neuron 5 pada huruf 'C' dengan akurasi 54%. Jika dilihat dari akurasi rata-rata pengenalan, kinerja terbaik adalah pada jumlah neuron 40 dengan akurasi rata-rata 78.4%. nilai rentang DS-B lebih kecil dibanding dengan DS-K disebabkan karena akurasi rata-rata DS-B rendah, selisih terkecil diperoleh 27.4% pada jumlah neuron 20 dan 35. Berpedoman pada hasil polah huruf 'C' menjadi pola dengan pengenalan terendah dari 4 huruf lainnya dari semua desain jumlah neuron yang digunakan dalam penelitian ini.

Setelah melakukan observasi secara medalam presisi kecil yang diterima oleh huruf 'C', sisa huruf 'C' yang tidak terkenali, ternyata terkenali sebagian kecil ke huruf 'D' dan dan sebagian besar ke huruf 'E', dapat dilihat pada Gambar 16 berikut:



Gambar 16 Prosentase pola huruf 'B' terkenali ke huruf 'D' dan 'E' pada DS-B *training* data

Hasil yang ditunjukkan pada grafik adalah hanya menampilkan prosentase huruf 'C' dikenali sebagai huruf 'D' dan 'E'. hasil di atas memberikan informasi bahwa pola huruf 'C' jika tidak dikenali sebagai dirinya sendiri maka dia lebih cendrung dikenali sebagai 'D' dan 'E', bahkan cendrung lebih banyak ke huruf 'E'. Pada akurasi tertinggi pada jumlah neuron didapatkan waktu 7 menit dan 44 detik dengan RMSE=0.26.

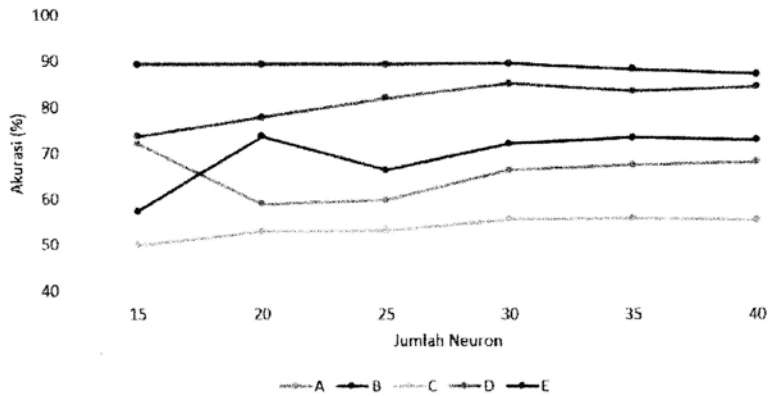
Tabel 9 Analisis Perbandingan JST DS-B pada 5 Pola Karakter pada sejumlah Variasi Neuron untuk *Testing* Data

Jumlah neuron *Hidden Layer*	A	B	C	D	E	Akurasi (Rata-rata)	Rentang	Waktu
	%	%	%	%	%	%	%	Menit.Detik
15	72.1	57.4	50.0	73.8	89.3	68.5	39.3	0.07
20	59.0	73.8	53.0	77.9	89.3	70.6	30.3	0.08
25	59.8	66.4	53.3	82.0	89.3	70.2	36.0	0.09
30	66.4	72.1	55.6	85.1	89.5	73.8	32.9	0.11
35	67.5	73.5	55.8	83.4	88.3	73.7	32.5	0.12
40	68.2	73.0	55.4	84.5	87.2	73.7	31.8	0.12

Keterangan : Rentang = $X_{max} - X_{min}$, X_{max} adalah nilai data kinerja tertinggi pada lima pola huruf

Hasil yang ditunjukkan pada Tabel 9 adalah penyajian hasil *testing* data dari JST-BP DS-B, hasil menunjukkan bahwa kinerja terbaik ditunjukkan pada jumlah neuron 30 pada pengenalan pola huruf 'D' dengan akurasi 89.5%, dan kinerja terendah terdapat pada jumlah neuron 5 huruf 'C' dengan presisi 50%. Dilihat dari rata-rata maka kinerja terbaik ditunjukkan pada desain jumlah neuron 35 dengan akurasi 73.8%, hasil tertinggi ini tidak bisa dikatakan efisien karena disamping nilainya tidak terlalu tinggi, juga tidak terjadi pengenalan secara merata antar huruf, karena nilai rentang masih diatas nilai rentang *training* data, rentang dengan selisih terendah saja pada angka 30.1% pada jumlah neuron 20. Pengenalan pada huruf 'C' mengikuti pola *training* data, nilai yang dihasilkan pada setiap desain jumlah neuron hanya berada pada presisi 55.8% ke bawah. Untuk komponen waktu didapatkan sangat kecil.

Hasil yang ditampilkan dari *training* data dan *testing* data, kedua mempunyai pola yang sama, pengenalan terendah ada pada pola huruf 'C' dan sisanya dikenali sebagai huruf 'D' dan 'E', jumlah neuron 30 didaulat sebagai desain jumlah neuron terbaik dari desain neuron lainnya, setelah jumlah neuron 30 kemungkinan menagalami *overfitting* sehingga kinerja tetap atau bisa jadi menurun, dan tidak terjadi pemerataan antara huruf dan yamg paling utama adalah bahwa DS-B belum mampu mengoptimasi secara baik kinerja dari JST akibat kurangnya ciri unik dari pola, terutama pola huruf 'C'.



Gambar 17 Hasil perbandingan pengenalan pola antar huruf *training* data DS-B

Hasil Gabungan Downsampling (Reduksi) Kolom dan Baris (DS-KB) Matriks

Hasil percobaan untuk DS-KB juga disajikan dalam dua tabel dan satu grafik sebagai pembanding hasil *training* data dan *testing* data, dan hasilnya sebagai berikut:

Tabel 10 Analisis Perbandingan JST DS-KB pada 5 Pola Karakter pada sejumlah Variasi Neuron untuk *Training* Data

Jumlah neuron *Hidden Layer*	A	B	C	D	E	Akurasi (Rata-rata)	Rentang	Waktu	RMSE
	%	%	%	%	%	%	%	Menit.Detik	
15	95.0	97.0	96.4	98.9	98.8	97.2	3.90	5.30	0.15
20	97.8	97.4	99.4	98.4	97.5	98.1	2.02	6.41	0.13
25	98.0	98.0	99.2	99.5	99.4	98.8	1.56	7.40	0.10
30	98.6	98.9	98.9	99.5	98.0	98.8	1.56	9.20	0.10
35	99.8	97.2	98.8	99.1	99.7	98.9	2.65	10.25	0.09
40	99.2	99.5	99.8	99.5	98.4	99.3	1.40	12.30	0.08

Keterangan : Rentang = $X_{max} - X_{min}$, X_{max} adalah nilai data kinerja tertinggi pada lima pola huruf

Hasil *training* data untuk DS-KB yang ditunjukkan Tabel 10 dengan 6 variasi jumlah neuron. Kinerja tertinggi diwakili oleh jumlah neuron 40 dengan nilai akurasi 99.8%, pengenalan terendah pada jumlah neuron 15 pada penegnalan huruf 'A' dengan akurasi 95%. Peningkatan jumlah neuron memberikan peningkatan akurasi dan tercatat pada jumlah neuron 40 dengan nilai akurasi tertinggi yaitu 99.3%. Begitu juga dengan nilai RMSE, dengan meningkatnya jumlah neuron, RMSE mengalami penurunan yang sangat tipis, kecuali pada jumlah neuron 25 dan 30, nilai RMSE-nya sama yaitu 0.10. Disamping akurasi rata-rata yang tinggi, didapatkan rentang (selisih) begitu tipis, rentang terendah yang diperoleh adalah 1.40% pada jumlah neurom 40. kemudian dari sisi waktu, sangat tidak bisa dipungkiri bahwa dengan bertambahnya jumlah neuron pada *hidden layer* maka membutuhkan waktu yang lama, jika dilihat dari hasil jarak waktu yang dibutuhkan antar jumlah neuron sekitar 1 menit dan beberapa detik.

Tabel 11 Analisis Perbandingan JST DS-KB pada 5 Pola Karakter pada sejumlah Variasi Neuron untuk *Testing* Data

Jumlah neuron *Hidden Layer*	A	B	C	D	E	Akurasi (Rata-rata)	Rentang	Waktu
	%	%	%	%	%	%	%	Menit.Detik
15	89.3	88.5	85.2	93.4	94.3	90.2	9.02	0.08
20	92.6	88.5	90.2	92.6	92.6	91.3	4.10	0.09
25	92.6	91.0	88.5	95.1	92.6	92.0	6.56	0.10
30	94.3	94.3	91.8	95.1	93.4	93.8	3.28	0.12
35	96.7	94.3	93.4	92.6	94.3	94.3	4.10	0.14
40	96.7	95.1	90.2	94.3	92.6	93.8	6.56	0.17

Keterangan : Rentang = $X_{max} - X_{min}$, X_{max} adalah nilai data kinerja tertinggi pada lima pola huruf

Tabel 11 adalah hasil *testing* data dari JST DS-KB. Kinerja terbaik ditunjukkan pada jumlah neuron 35 dan 40 dengan hasil yang sama dengan nilai akurasi yang sama 96.7% dan juga pada pola yang sama pada huruf 'A', kemudian kinerja terendah terdapat pada jumlah neuron 5 pada pengenalan pola huruf 'C' dengan akurasi 85.2%. Peningkatan rata-rata pada jumlah neuron mulai dari jumlah neuron 15 (90.2%) sampai dengan jumlah neuron 35 (94.3%), dan pada jumlah neuron 40 mengalami penurunan. Kemudian rentang yang dihasilkan memang lebih besar dari hasil *training* data, akan tetapi masih tergolong kecil dengan angka selisih terendah 3.28% adalah angka yang tergolong kecil. Selanjutnya dilihat dari sisi

waktu, dengan bertambahnya jumlah neuron juga memberikan penambahan waktu, dan yang sama hanya pada jumlah neuron 15 dan 20.

Membandingkan hasil *training* data (Tabel 10) dan *testing* data (Tabel 11), Pada hasil *training* data, dari 6 variasi jumlah neuron secara rata-rata yang tertinggi kinerjanya adalah jumlah neuron 40, akan tetapi hasil tersebut tidak mengindikasikan bahwa jumlah neuron inilah yang efisien, untuk melihat desain mana yang paling efektif harus dibandingkan dengan hasil *testing* data, pada *testing* data kinerja terbaik ditunjukkan oleh 35 dengan akurasi 94.3%, artinya bahwa desain dengan jumlah neuron 40 menjadi tidak terlalu efisien dibanding dengan jumlah neuron 35 pada *testing* data. Untuk tahap terakhir dalam menentukan desain yang mana paling efesien maka dianalisa dengan rentang (pemerataan pengenalan antar pola).

Kenaikan Akurasi Pola Huruf 'B' dan 'C' pada DS-KB

Dua pola yang terkenali sangat rendah sebelumnya, yaitu pola huruf 'B' pada DS-K dan pola huruf 'C' pada DS-B, untuk melihat prosentasenya pada DS-KB dapat dilihat pada tabel-tabel berikut ini:

Tabel 12 Kenaikan Prosentase pengenalan untuk pola huruf 'B' pada DS-KB untuk hasil *training* data dan *testing* data

Jumlah neuron *Hidden Layer*	*Training* Data - Huruf 'B' (%)					*Testing* Data - Huruf 'B' (%)				
	DS-K	DS-B	Rata2	DS-KB	Kenaikan	DS-K	DS-B	Rata2	DS-KB	Kenaikan
15	60.1	65.4	62.8	97.0	***34.3***	60.7	57.4	59.0	88.5	***29.5***
20	58.7	80.2	69.5	97.4	***27.9***	57.4	73.8	65.6	88.5	***23.0***
25	66.0	72.7	69.4	98.0	***28.6***	62.3	66.4	64.3	91.0	***26.6***
30	71.7	77.5	74.6	98.9	***24.3***	62.3	72.1	67.2	94.3	***27.1***
35	68.5	76.5	72.5	97.2	***24.7***	63.1	73.5	68.3	94.3	***26.0***
40	68.7	78.6	73.6	99.5	***25.9***	60.7	73.0	66.8	95.1	***28.3***

Pada Tabel 12, memberikan informasi prosentase kenaikan pengenalan huruf 'B' pada algoritma DS-KB, pengenalan pada algoritme DS-K terhitung rendah, dan pada algoritme DS-B mengalami kenaikan, penggabungan keduanya pada algoritme DS-KB memberikan efek yang cukup baik, baik dari sisi *training* data maupun pada *testing* data, kenaikan pengenalan pada huruf 'B' pada DS-KB cukup tinggi, dilihat dari kecilnya rata-rata, didapat rata-rata akurasi tertinggi 74.6% (*training* data) pada jumlah neuron 30 dan nilai akurasi 68.3% (*tetsing* data), rendahnya akurasi rata-rata disebabkan oleh pengenalan huruf 'B' pada DS-K cukup rendah.

Tabel 13 Kenaikan Prosentase pengenalan untuk pola huruf 'C' pada DS-KB untuk hasil *training* data dan *testing* data

Jumlah neuron *Hidden Layer*	*Training* Data - Huruf 'C'					*Testing* Data - Huruf 'C' (%)				
	DS-K	DS-B	Rata2	DS-KB	Kenaikan (%)	DS-K	DS-B	Rata2	DS-KB	Kenaikan
15	96.6	54.0	75.3	96.4	***21.1***	86.9	50.0	68.4	85.2	***16.8***
20	94.9	65.5	80.2	99.4	***19.2***	86.1	59.0	72.5	90.2	***17.6***
25	95.0	61.6	78.3	99.2	***20.9***	89.3	53.3	71.3	88.5	***17.2***
30	96.1	63.6	79.8	98.9	***19.1***	90.2	55.6	72.9	91.8	***18.9***
35	96.8	62.8	79.8	98.8	***19.0***	90.2	55.8	73.0	93.4	***20.5***
40	92.4	63.5	77.9	99.8	***21.9***	79.5	55.4	67.5	90.2	***22.7***

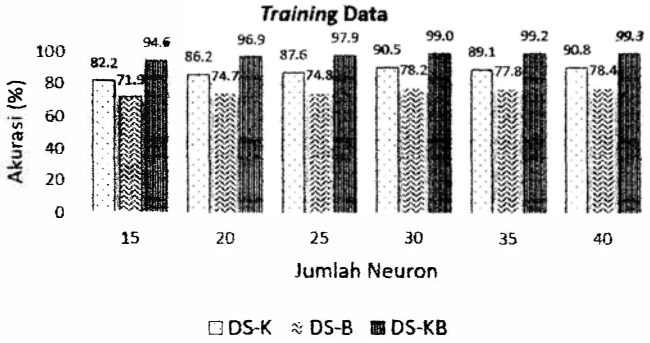
Pada Tabel 13, memberikan informasi prosentase kenaikan pengenalan huruf 'C' pada algoritme DS-KB, pengenalan pada algoritme DS-K terhitung tinggi, dan pada algoritme DS-B pengenalan sangat rendah, penggabungan keduanya pada algoritme DS-KB memberikan efek yang cukup baik.

Perbandingan hasil DS-K, DS-B, dan DS-KB dan memilih Teroptimasi

Perbandingan dibawah ini meliputi 3 komponen, yaitu Akurasi, Rentang, dan Waktu, ketiga hasil dari komponen ini yanga akan dijadikan sebahai bahan analisa untuk memilih algoritme teroptimasi.

1) Akurasi

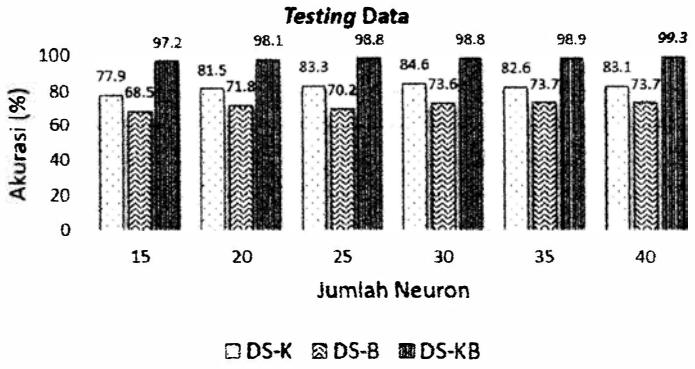
Semakin tinggi nilai akurasi amak semakin memberikan sinyalemen bahwa sebuah algoritme berkinerja baik. Hasil yang ditampilkan dibawah ini adalah sebuah tabel hasil perbandingan antara JST DS-K, DS-B. dan DS KB.



Gambar 18 Perbandingan Akurasi DS-K, DS-B, dan DS-KB pada hasil *trainig* data

Gambar 18 adalah menunjukkan hasil tingkat akurasi 3 algoritme, yaitu; DS-K, DS-B, dan DS-KB pada *training* data. DS-KB memberikan akurasi tertinggi

dibanding dengan 2 lagoritme lainnya, penggabungan dua algoritme memberikan efek peningkatan yang cukup signifikan, akurasi tertinggi pada DS-KB dengan tingkat akurasi 99.3% pada jumlah neuron 40.



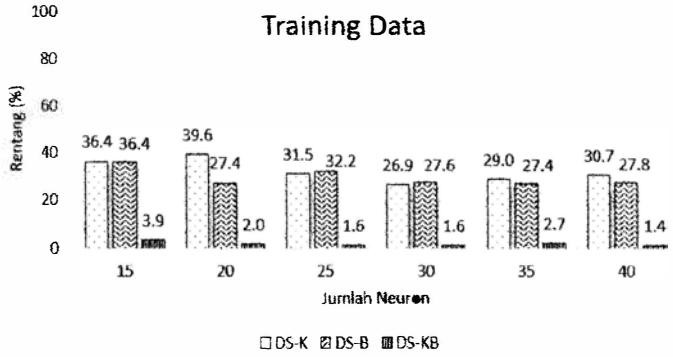
Gambar 19 Perbandingan Akurasi DS-K, DS-B, dan DS-KB pada hasil *testing* data

Gambar 19 adalah menunjukkan hasil tingkat akurasi 3 algoritme, yaitu; DS-K, DS-B, dan DS-KB pada *testing* data. DS-KB memberikan akurasi tertinggi dibanding dengan 2 lagoritme lainnya, walaupun pada jumlah neuron 35 ke bawah ada penurunan prosentase dibanding dengan hasil *training* data, akan tetapi pada jumlah neuron 40, hasil akurasi yang kebetulan sama dengan hasil DS-KB *training* data dengan akurasi 99.3%.

Garafik yang ditampilkan pada Gambar 18 dan Gambar 19 mendaulat algoritme DS-KB sebagai algoritme terbaik dibanding dengan alogoritme DS-K dan DS-KB

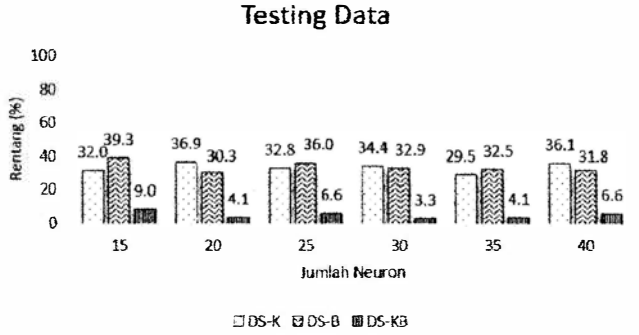
2) Rentang

Rentang (*Range*) seperti telah dijelaskan sebelumnya adalah selisih antara data dengan nilai terbesar (x_{max}) dengan data nilai yang terkecil (x_{min}) atau dalam formulasi ilmu statistik dikenal dengan $R = x_{max} - x_{min}$, x_n adalah nilai presisi setiap pola yang dalam penelitian ada 5, nilai presisi 'A', 'B', 'C', 'D', dan 'E', semakin rendah nilai rentang (selisih) yang dihasilkan maka semakin baik presisi pengenalan dan pemertaan antar pola.



Gambar 20 Perbandingan Rentang DS-K, DS-B, dan DS-KB pada hasil *testing* data

Membandingkan Rentang ketiga algoritme yang ditunjukkan pada Gambar 20, memberikan gambaran jelas, DS-K dan DS-B menunjukkan nilai rentang diatas 20%, nilai tersebut menjelaskan bahwa pada kedua algoritme tersebut tidak terjadi pemerataan pengenalan antar pola, terjadi selisih yang besar antar pola terekanali dengan akurasi tinggi dengan pola yang terkenali dengan akurasi rendah, sementara hasil yang ditunjukkan untuk pada DS-K menghasilkan rentang yang sangat tipis dan menjelaskan bahwa algoritme gabungan atau DS-KB memberikan kinerja yang cukup baik, ditemukan 3 jumlah neuron yang cukup tipis, yaitu; jumlah neuron 25 denga rentang 1.6%, jumlah neuron 30 dengan rentang 1.6%, dan jumlah neuron 40 dengan rentang 1.4%.



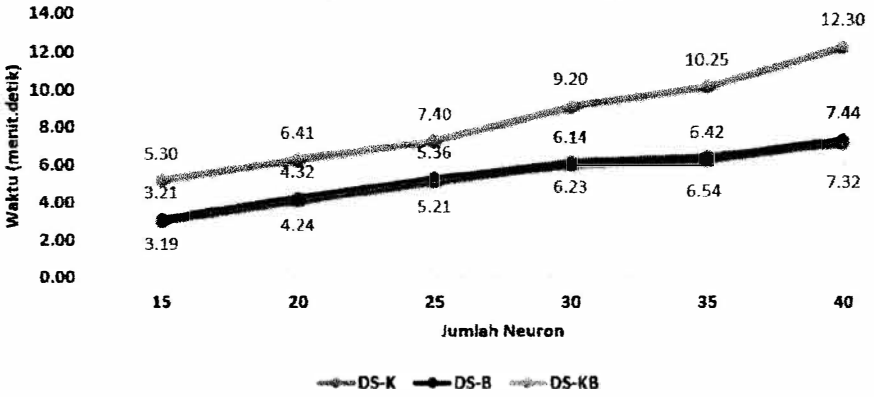
Gambar 21 Perbandingan Rentang DS-K, DS-B, dan DS-KB pada hasil *testing* data

Untuk hasil *testing* data pada Gambar 21, memang memberikan nilai rentang diatas dari nilai rentang *training* data, terutama pada algoritme DS-K dan DS-B, sementara kenaikan nilai rentang pada DS-KB juga terjadi cukup tinggi terutama pada jumlah neuron 15 dengan nilai selisih 9%, yang cukup baik adalah pada jumlah neuron 30 dengan nilai rentang 3.3%.

Hasil yang ditampilkan oleh Gambar 20 dan Gambar 21, menggambarkan jelas bahwa DS-KB tetapi menjadi algoritme terbaik dibanding dengan alogoritme DS-K dan DS-B.

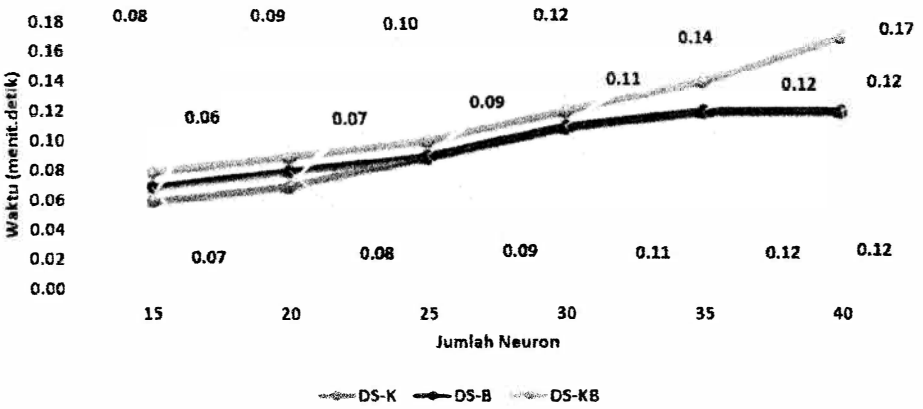
3) *Waktu*

Waktu yang ditampilkan pada grafik dibawah adalah waktu yang digunakan selama *training* data dan *testing* data.



Gambar 22 Perbandingan Waktu DS-K, DS-B, dan DS-KB pada hasil *training* data

Gambar 22 menyajikan grafik perbandingan waktu yang dibutuhkan pada saat *training* data pada 3 algoritma. DS-K dan DS-B memberikan waktu yang cukup singkat dibanding dengan DS-KB, alasan yang sangat logis yang bisa diberikan adalah bahwa DS-K dan DS-B memiliki data *input* pada *input layer* lebih sedikit dibanding dengan DS-KB, DS-K dan DS-B hanya memiliki 50 neuron pada *input layer*, sedangkan DS-KB 2 kali lipat dari keduanya, yaitu 100 neuron pada *input layer*. Misalnya pada jumlah neuron 15 dan 40 pada *hidden layer*, pada jumlah neuron 15, DS-K dan DS-B waktu yang dibutuhkan hampir sama kisaran 3 menit, akan tetapi pada DS-KB membutuhkan waktu 5 menit dan 30 detik, sementara pada jumlah neuron 40, DS-K dan DS-B membutuhkan waktu sekitar 7 menit, sementara pada DS-KB mengalami lonjakan sekitar 5 menit atau tepatnya 12 menit dan 30 detik.



Gambar 23 Perbandingan Waktu DS-K, DS-B, dan DS-KB pada hasil *testing* data

Pola yang sama yang ditampilkan Gambar 28 menyajikan grafik perbandingan waktu yang dibutuhkan pada saat *testing* data. DS-K dan DS-B memberikan waktu yang cukup singkat dibanding dengan DS-KB, Misalnya pada jumlah neuron 15 dan 40 pada *hidden layer*, pada jumlah neuron 15, DS-K dan DS-B waktu yang dibutuhkan hampir sama kisaran 6 detik pada DS-K dan 7 detik pada DS-B, akan tetapi pada DS-KB membutuhkan waktu 8 detik, sementara pada jumlah neuron 40, DS-K dan DS-B membutuhkan waktu sekitar 12 detik, sementara pada DS-KB mengalami lonjakan sekitar 5 detik.

Hasil yang ditampilkan oleh Gambar 27 dan Gambar 28, menggambarkan jelas bahwa DS-K dan DS-B jauh lebi cepat dibanding dengan DS-KB.

4) Memilih Teroptimasi

Menganalisa hasil dengan tiga komponen, yaitu; Akurasi, Rentang, dan Waktu, DS-KB memiliki 2 keunggulan yaitu tingkat akurasi tinggi dan rentang yang cukup tipis baik dari *training* data maupun *testing* data, sedangkan DS-K dan DS-B hanya unggul dari sisi waktu, dua keunggulan yang dimiliki oleh DS-KB memberikan cukup alasan untuk memilih algoritme ini menjadi yang teroptimasi, Jika dengan alasan waktu yang dibutuhkan algoritme DS-KB lebih banyak dibanding dengan DS-K dan DS-KB, Apa artinya sebuah waktu yang sangat singkat jika tidak memiliki akurasi yang tinggi dan rentang (*range*) yang tipis, maka dengan alasan tersebut mengapa waktu menjadi alat analisa ketiga setelah akurasi dan rentang, kemudian waktu yang dibutuhkan oleh DS-KB juga tidak terlalu lama dibandingkan dengan pelatihan yang tanpa menggunakan *downsampling* (reduksi) matriks citra (normal).

Memilih Jumlah Neuron Hidden Layer sebagai teroptimasi pada DS-KB

Sekarang dipersempit pada hasil DS-KB, telah dijelaskan sebelumnya bahwa memilih desain jumlah neuron yang efisien bukan hanya melihat akurasi hasil *training* data, akan tetapi harus dipadukan dengan hasil *testing* data, apa artinya sebuah akurasi yang tinggi ketika *training* data akan tetapi menghasilkan kinerja yang rendah pada saat *testing* data. Perhatikan Tabel 17 berikut:

Tabel 14 Perbandingan Kinerja DS-KB antara *training* data dan *testing* data dan rentang terbesar dengan rentang terkecil.

Jumlah neuron *Hidden Layer*	Training Data				Testing Data			
	%	Maks	Min	Rentang (Maks - Min)	%	Maks	Min	Rentang (Maks - Min)
15	97.2	98.9	95.0	3.9	90.2	94.3	85.2	9.0
20	98.1	99.4	97.4	2.0	91.3	92.6	88.5	4.1
25	98.8	99.5	98.0	1.6	92.0	95.1	88.5	6.6
30	98.8	99.5	98.0	1.6	93.8	95.1	91.8	3.3
35	98.9	99.8	97.2	2.7	94.3	96.7	92.6	4.1
40	99.3	99.8	98.4	1.4	93.8	96.7	90.2	6.6

Keterangan : Maks adalah kinerja tertinggi pada lima pola huruf
: Min adalah kinerja terendah pada lima pola huruf

Sekarang tinggal memilih mana desain yang lebih optimasi dari DS-KB. Mekanimse yang digunakan adalah memilih akurasi tertinggi dengan membandinkan antara *training* data dan *testing* data, selanjutnya melihat rentang, lalu kemudian waktu.

Pada Tabel 14, didapatkan empat jumlah neuron kinerja tinggi yaitu 25, 30, 35, dan 40, dengan rentang tipis pada jumlah neuron 25, 30, dan 40, dan waktu mulai dari 7 menit dan 40 detik hingga 12 menit dan 30 detik, artinya ketiga jumlah neuron tersebutlah yan akan dibandingkan dengan *testing* data. Pada *testing* data, dilakukan seleksi juga dengan menyeleksi jumlah neuron dengan akurasi baik yang juga dimulai dari jumlah neuron 20 hingga jumlah neuron 40 dengan waktu dimulai dari 9 detik hingga 17 detik, artinya ada 5 jumlah neuron pada *testing* data dengan waktu bervariasi tapi masih dalam hitungan detik, akan tetapi dari 5 jumlah neuron tersebut hanya 3 yang mempunyai rentang yang tipis yaitu 20, 30, dan 35.

Sekarang dipersempit area pemilihan untuk penyeleksiannya, dari *training* data sudah di seleksi 3 jumlah neuron dengan pertimbangan 3 komponen, yaitu jumlah neuron 25, 30, dan 40, sementara pada *testing* data telah diseleksi 3 jumlah neuron, yaitu 20, 30, dan 35. Kemudian eliminasi dari keduanya dengan menemukan titik temu dari keduanya, dan yang menjadi titik temunya adalah jumlah neuron 30, jadi pada penelitian ini algoritme DS-KB untuk jumlah neuron teroptimasi ada pada jumlah neuron 30 dengan akurasi pada presisi 98.8%, rentang (pemerataan antar pola) – 3.3%, waktu 9 menit dan 20 detik, dengan RMSE=0.1.

Hasil Downsampling (DS) dengan variasi Iterasi

Hasil dibawah adalah hasil ujicoba DS-KB dengan variasi iterasi, jumlah variasi iterasi yang diamti adalah 8 variasi iterasi, yaitu iterasi 500, 1 000, 1 500, 2 000, 2 500, 3 000, 3 500, dan 4 000. Pada sub bab sebelumnya telah ditetapkan DS-KB sebagai algoritme yang paling optimasi dengan desain jumlah neuron 30, kemudian pada sub bab akan menganalisa lebih dalam, mungkin pertanyaan yang bisa dimunculkan adalah pada iterasi keberapa DS-KB pada jumlah neuron 30 memberikan hasil yang optimal?

Tabel 15 Hasil Algoritme DS-KB dengan variasi iterasi pada *training* Data

Iterasi	A	B	C	D	E	Akurasi	Maks	Min	Rentang	Waktu	RMSE
	%	%	%	%	%	%	(A,B,C,D,E)	(A,B,C,D,E)	(Maks-Min)		
500	83.8	76.2	91.3	78.0	73.8	80.6	91.3	73.8	17.5	1.11	0.39
1000	95.0	82.7	97.4	92.1	90.0	91.4	97.4	82.7	14.6	2.14	0.29
1500	97.5	89.7	98.3	97.0	94.6	95.4	98.3	89.7	8.6	3.11	0.23
2000	98.4	95.0	98.8	99.1	97.2	97.7	99.1	95.0	4.1	4.16	0.18
2500	98.4	97.8	98.8	99.4	97.7	98.4	99.4	97.7	1.7	5.40	0.15
3000	98.6	98.6	98.8	99.4	98.0	98.7	99.4	98.0	1.4	6.47	0.13
3500	98.6	98.8	98.8	99.4	98.0	98.7	99.4	98.0	1.4	7.56	0.10
4000	98.6	98.9	98.9	99.5	97.9	98.8	99.5	97.9	1.6	9.20	0.08

Pada Tabel 15 menampilkan hasil *training* data dengan variasi iterasi, pengamatan dimulai dari iterasi 500 dan dipantau hingga iterasi 4 000 dengan kelipatan 500. Jika dilihat dari sisi rata-rata maka penampakan yang bagus dimulai dari iterasi 1 000 dengan presisi 91.4%, namu jika melihat waktu ada kenaikan ± 1 menit tiap variasi iterasi kecuali pada iterasi 4 000, dari sisi rentang, rentang mulai sangat kecil mulai dari iterasi 2 500 dengan rentang 1.7%, artinya bahwa algoritme DS-KB sebagai nilai input untuk JST-BP memberikan kinerja yang baik, sedangkan untuk RMSE mulai dari iterasi 500 dengan nilai 0.39 dan terus memperkecil diri hingga iterasi 4 000 dengan RMSE=0.08 dengan penurunan yang tak begitu tajam.

Berdasarkan hasil di atas dengan melihat akurasi, waktu, dan rentang maka pada iterasi 2 500 adalah iterasi yang teroptimal disebabkan pada iterasi itu akurasi sudah mencapai 98.4% dengan waktu 5 menit dan 40 detik, dan dari sisi rentang atau pemerataan antar pola sudah sangat tipis dengan angka 1.7% dan dengan tingkat error pada RMSE=0.15.

5 SIMPULAN DAN SARAN

Simpulan

Penelitian ini membangun model ekstraksi ciri untuk optimasi pengenalan pola JST, model yang dibagun adalah *downsampling* (reduksi) matriks citra dengan tiga metode, yaitu; *Downsampling* (reduksi) Kolom Matriks Citra (DS-K), *Downsampling* (reduksi) Baris Matriks Citra (DS-B), dan *Downsampling* (reduksi) Kolom dan Baris Matriks Citra (DS-KB). Ketiga metode yang dibangun kemudian akan dibandingkan untuk melihat mana yang lebih optimasi dalam pengenalan pola JST. Data sampel yang digunakan untuk pengujian model adalah tulisan tangan huruf kapital 'A', 'B', 'C', 'D' dan 'E' dengan studi kasus Lembar Jawaban Ujian (LJU) Perguruan Tinggi.

Percobaan yang dilakukan dengan variasi jumlah neuron *hidden layer* pada JST-BP dengan input dari 3 algoritme DS, yaitu DS-K, DS-B, dan DS-KB. DS-KB menghasilkan akurasi tertinggi terutamapada jumlah neuron 40 dengan akurasi 99.3% dengan waktu *training* adalah 12 menit dan 30 detik, dengan tingkat *error* dengan nilai RMSE=0.08, dan pada *testing* data akurasi tertinggi adalah pada jumlah neuron 35 dengan akurasi 94.3%, rentang (selisih) rendah, terjadi pemerataan pengenalan antar pola dengan rentang 3.3% pada jumlah neuron 30 pada *testing* data.

Penggabungan DS-K dan DS-B, menjadi DS-KB sebagi model yang terbaik untuk digunakan dengan melihat hasil *training* data dan *testing* data dengan pertimbangan akurasi, rentang dan waktu maka didapatkan bahwa jumlah neuron 30 adalah desain yang optimal.

Saran

Penelitian ini memiliki beberapa keterbatasan, berikut beberapa saran yang dapat dilakukan pada penelitian selanjutnya:

1. Bahwa ujicoba kita hanya pada huruf kapital 'A', 'B', 'C', 'D', dan 'E', maka perlu diujicoba dengan menggunakan semua huruf termasuk angka.
2. Pada pengenalan pola pada JST, belum banyak menggunakan algoritme optimasi, misalnya optimasi *random* nilai bobot dan optimasi.

Empat poin diatas bisa menjadi konsentrasi penelitian berikutnya untuk bisa mengoptimalisasi penelitian *downsampling* (reduksi) matriks citra pada algoritme DS.

DAFTAR PUSTAKA

Achmad B, Firdausy K. 2005. *Teknik Pengolahan Citra Digital Menggunakan Delphi*. Ed ke-1. Yogyakarta (ID): Ardi Publishing.

Asht S, Dass R. 2012. Pattern recognition techniques: A review. *International Journal of Computer Science and Telecommunications*. 3(8):5.

Basu JK, Bhattacharyya D, Kim TH. 2010. Use of artificial neural network in pattern recognition. *International Journal of Software Engineering and Its Applications*. 4(2):12.

Belawati T. 2000. Enhancing learning in distance education through the world wide web. *Pendidikan Terbuka dan Jarak Jauh*. 1(1):10.

Bhat M. 2014. Digital image processing. *International Journal of Scientific & Technology Research*. 3(1):5. doi:2277-8616.

Bhargava N, kumawat A, Bhargava R. 2014. Threshold and binarization for document image analysis using otsu's algorithm. *International Journal of Computer Trends and Technology (IJCTT)*. 17(5):4.

Boger Z, Guterman H. 1997. Knowledge extraction from artificial neural networks models. Di dalam: *IEEE Systems, Man and Cybernetics Conference*; Oktober 1997; Orlando. Orlando (US): IEEE Systems. hlm 12.

Chen H, Huang X, Cheng P, Xu Y. 2013. Off-line handwritten chinese character recognition based on GA optimization BP neural network. *International Conference on Information, Business and Education Technology (ICIBIT)*. 1(1):4.

Fausett LV. 1994. *Fundamental of Neural Network: Architecture, Algorithm and Application*. Ed ke-1. New Jersey (US): Prentice- Hall.

Hagan MT, Demuth HB, Beale MH, Jesus OD. 2002. *Neural Network Design*. Ed ke-1. China (HK): China Machine Press.

Jadhav U, Shetty A. 2016. Effect of varying neurons in the hidden layer of neural network for simple character recognition. *International Journal on Recent and Innovation Trends in Computing and Communication*. 4(6):4. doi:2321-8169.

J.Pradeep, E.Srinivasan, S.Himavathi. 2011. Diagonal based feature extraction for handwritten alphabets recognition system using neural network. *International Journal of Computer Science & Information Technology (IJCSIT)*. 3(1):12.

Kacprzyk J, Pedrycz W. 2014. *Computational Intelligence*. New York (US): Springer.

Karsoliya S. 2012. Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture. *International Journal of Engineering Trends and Technology*. 3(6):4. ISSN:2231-5381.

Kusumadewi S. 2003. *Artificial Intelligence (Teknik dan Aplikasinya)*. Yogyakarta (ID): Graha Ilmu.

Muntasa A. 2015. *Pengenalan Pola*. Ed ke-1. Yogyakarta (ID): Graha Ilmu.

Panchal G, Ganatra A, Kosta Y, Panchal D. 2011. Behaviour Analysis of Multilayer Perceptrons with Multiple Hidden Neurons and Hidden Layers. *International Journal of Computer Theory and Engineering*, 3(2):4. ISSN:1793-8201.

Perwej Y, Chaturvedi A. 2011. Neural networks for handwritten english alphabet recognition. *International Journal of Computer Applications*. 20(7):5.

Putra D. 2010. *Pengolahan Citra Digital*. Ed ke-1. Westriningsih, editor. Yogyakarta (ID): Andi Offset.

Sarangi PK, Ravulakollu KK. 2014. Feature extraction and dimensionality reduction in pattern recognition using handwritten odia numerals. *Middle-East Journal of Scientific Research*. 22(10):6. doi:10.5829/idosi.mejsr.2014.22.10.21487.

Sarangi PK. 2013. Recognition of handwritten odi a numerals using artificial intelligence techniques. *The International Journal of Computer Science & Application S (TIJCSA)*. 2(2):8. ISSN:2278-1080.

Sharma P, Malik N, Akhtar N, Rahul, Rohilla H. 2013. Feedforward neural network: A review. *International Journal of Advanced Research in Engineering and Applied Sciences*. 2(10):10. ISSN:2278-6252.

S S, kumar PS. 2014. Image contrast enhancement using histogram equalization techniques: review. *International Journal of Advances in Computer Science and Technology*. 3(3):10.

Shih Y. 1994. *Neuralyst User's Guid*. Ed ke-4. Farley S, editor. United States of America (US): Cheshire Engineering Corpora.

Sutojo T, Mulyanto E, Suhartono V. 2011. *Kecerdasan Buatan*. Ed ke-1. W BR, editor. Yogyakarta (ID): Andi, Udinus.

Theodoridis S, Koutroumbas K. 2003. *Pattern Recognition*. Ed ke-2. Amerika (US): Elsevier Academic Press.

Vala MHJ, Baxi A. 2014. A review on otsu image segmentation algorithm. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*. 2(2):3.

Yang W, Xu L, Chen X, Zheng F, Liu Y. 2014. Chi-squared distance metric learning for histogram data. *Mathematical Problems in Engineering*. 2015:12.

Yoo SS, Kim YT, Youk SJ, Kim JH. 2006. Adaptive-binning color histogram for image information retrieval. *International Journal of Multimedia and Ubiquitous Engineering*. 1(4):9.

RIWAYAT HIDUP

Kani, dilahirkan di Sempang pada tanggal 14 Juni 1978, sebuah dusun yang sangat terpencil di daerah Pinrang, Sulawesi Selatan, lahir dari seorang ibu bernama Hj. Tika dan ayah bernama H. Launggu, 9 bersaudara laki-laki semua dan sebagai anak bungsu. Riwayat pendidikan adalah MAN 2 Parepare, S1 Teknik Informatika di Universitas Muslim Indonesia di Makassar, saat ini bekerja di Universitas Terbuka Pusat di Pondok Cabe, Tangerang Selatan, Banten.