Pengenalan Tipe Tas Tangan Wanita Pada Citra Digital Menggunakan Jaringan Syaraf Tiruan Perambatan Balik

Edwin Kurniawan, Universitas Ciputra, UC Town Citraland, Surabaya 60219
Nehemia Sugianto, Universitas Ciputra, UC Town Citraland, Surabaya 60219

ABSTRAK

Industri *fashion* selalu berkembang dengan pesat dimana ditandai dengan munculnya banyak merek baru, tipe atau desain setiap harinya. Salah satu produk *fashion* adalah produk tas. Tas bisa berupa tas tangan, tas ransel, tas *pouch* ataupun tas lainnya. Dengan selalu munculnya tipe baru pada produk tas, maka makin bertambah banyak jumlah tipe tas dari tiap merek yang ada. Kondisi yang demikian membuat masyarat agak susah untuk menghafal tipe tas tersebut dan di mana mereka bisa membeli tas tersebut. Tujuan dari penelitian ini adalah menghasilkan sebuah sistem pengenalan tas tangan menggunakan jaringan syaraf tiruan perambatan balik. Penelitian hanya terbatas pada pengenalan tipe tas wanita yang berupa tas tangan dan hanya pada tipe tas tertentu. Proses pengenalan tas tangan terdiri dari beberapa tahap yaitu tahap pengambilan gambar, tahap *pre*-processing, tahap ekstraksi fitur, tahap klasifikasi. Tahap pengambilan gambar dilakukan menggunakan kamera dimana gambar yang didapatkan menjadi data masukan sistem. Tahap *pre-processing* merupakan tahap untuk mengolah gambar tersebut agar memudahkan proses ekstraksi fitur sehingga didapatkan fitur-fitur dengan hasil yang maksimal. Tahap ekstraksi fitur adalah tahap untuk mengekstraksi fitur dari sebuah gambar sehingga didapatkan fitur-fitur yang nantinya digunakan sebagai data masukan pada proses klasifikasi untuk mengenali tas tangan pada gambar tersebut. Tahap klasifikasi adalah tahap untuk mengenali tipe tas tangan menggunakan jaringan syaraf tiruan perambatan balik berdasarkan fitur-fitur yang didapatkan pada tahap sebelumnya.
Jaringan syaraf tiruan yang digunakan memiliki tiga lapisan yaitu lapisan masukan dimana terdiri dari 6 neuron, lapisan tersembunyi dimana terdiri dari 10 neuron dan lapisan keluran dimana terdiri dari 8 neuron. Pelatihan jaringan syaraf tiruan menggunakan algoritma peramabatan balik dengan nilai laju pelatihan sebesar 0,1 momentum sebesar 0,8. Pelatihan dilakukan menggunakan 414 data dengan nilai *epoch* sebesar 1.000 *epoch* dan nilai *Mean Squared Error (MSE)* sebesar 0.02. Pelatihan mencapai tingkat akurasi klasifikasi sebesar 71.1% dimana pelatihan berhenti pada *epoch* ke 980 dengan nila *MSE* sebesar 0.019641. Uji coba dilakukan menggunakan 28 data. Uji coba menunjukkan bahwa tingkat akurasi yang dihasilkan adalah 78,57%.

Kata kunci: Pengenalan Obyek, Ekstraksi Fitur, Klasifikasi, Pengolahan Citra Digital, Jaringan Syaraf Tiruan Perambatan Balik, OpenCV

1. PENDAHULUAN

Industri *fashion* selalu berkembang dengan pesat dimana ditandai dengan munculnya banyak merek baru, tipe atau desain setiap harinya. Salah satu produk *fashion* adalah produk tas. Tas bisa berupa tas tangan, tas ransel, tas *pouch* ataupun tas lainnya. Dengan selalu munculnya tipe baru pada produk tas, maka makin bertambah banyak jumlah tipe tas dari tiap merek yang ada. Kondisi yang demikian membuat masyarat agak susah untuk menghafal tipe tas tersebut dan di mana mereka bisa membeli tas tersebut. Tujuan dari penelitian ini adalah menghasilkan sebuah sistem pengenalan tas tangan menggunakan jaringan syaraf tiruan perambatan balik. Penelitian hanya terbatas pada pengenalan tipe tas wanita yang berupa tas tangan dan hanya pada tipe tas tertentu.

2. FITUR

Fitur-fitur tersebut akan digunakan sebagai data masukan pada proses klasifikasi untuk mengenali obyek tersebut.

Pemilihan fitur bergantung pada sifat obyek itu sendiri (seperti warna, tekstur, bentuk, ukuran, pergerakan dan sebagainya) dan kondisi lingkungan dari obyek yang akan dikenali (seperti kondisi latar belakang obyek, kehadiran obyek lain yang tidak diinginkan, iluminasi dan sebagainya). Ukuran, warna dan bentuk adalah fitur-fitur yang paling sering digunakan.

Fitur global *(global features)* adalah karakterisistik sebuah wilayah *(region)* dari gambar seperti luas *(area)*, keliling *(perimeter)*, *fourier descriptor*, *moments*. Fitur global didapatkan dengan cara mempertimbangkan semua titik dalam sebuah wilayah atau hanya pada titik-titik tersebut pada garis batas pada sebuah wilayah (Mingqiang, Kidiyo, & Josepj, 2008).

Fitur lokal *(local features)* biasanya berada di garis batas dari sebuah obyek atau mewakili sebuah area kecil yang dapat dibedakan dari sebuah wilayah. Fitur lokal yang sering digunakan adalah lengkungan *(curvature)*, segmen batas *(boundary segments)*, dan pojok *(corner)*. Lengkungan *(curvature)* atau properti yang berkaitan biasanya digunakan sebagai fitur lokal. Lengkungan bisa berupa lengkungan di garis batas atau dapat dihitung di sebuah luasan. Titik kelengkungan tingkat tinggi biasanya disebut pojok *(corner)* (Yang, 2008).

Fitur relasi *(relational features)* didasarkan pada posisi relatif dari entiti-entiti yang berbeda, baik wilayah, garis luar yang tertutup atau fitur lokal. Fitur-fitur ini biasanya melibatkan jarak antara fitur-fitur dan ukuran orientasi yang relatif. Fitur ini biasanya sangat berguna dalam menentukan obyek-obyek gabungan dengan menggunakan banyak wilayah atau fitur lokal dalam gambar (Yang, 2008).

2.1. Pemilihan Fitur

Pemilihan fitur yang tepat akan mengurangi kompleksitas komputasi (karena melibatkan faktor konsumsi waktu dan memori) dan menghasilkan tingkat akurasi pengenalan yang baik. Semakin banyak tipe tas tangan yang dapat dikenali oleh sistem dan perbedaan yang tipis antara tipe tas tangan yang satu dengan tipe tas tangan yang lain, maka dibutuhkan lebih banyak fitur agar dapat mengenali tipe tas tangan dengan tepat.

Ada dua fitur sederhana dalam mengenali tas tangan yang dapat digunakan dalam menentukan fitur-fitur yang akan diekstrak yaitu bentuk obyek (*shape*) dan tekstur obyek (*texture*). Fitur bentuk obyek dapat dinyatakan dengan *circularity* dan *rectangularity*. Warna tas tangan tidak dianggap sebagai fitur dikarenakan sebuah tipe tas tangan memiliki lebih dari satu warna tas tangan sehingga tidak mempengaruhi tipe tas tangan.

2.2. Ekstraksi Fitur

Setelah menentukan fitur-fitur yang akan digunakan dalam mengenali tas tangan dari sebuah gambar, maka dilakukan proses ekstraksi fitur dari sebuah gambar sehingga didapatkan fitur-fitur yang nantinya digunakan sebagai data masukan pada proses klasifikasi untuk mengenali tas tangan pada gambar tersebut.

Sebelum dilakukan proses ekstraksi fitur, dilakukan proses *pre-processing* untsuk memudahkan proses ekstraksi fitur sehingga didapatkan fitur-fitur dengan hasil yang maksimal. Proses *pre-processing* terdiri dari beberapa langkah yaitu proses konversi gambar dari *RGB* ke *grayscale*, proses segmentasi gambar, proses deteksi garis tepi, proses dilasi dan proses erosi.

- Proses konversi gambar dari *RGB* ke *grayscale*
  Proses ini bertujuan untuk mengurangi kompleksitas komputasi gambar dalam proses ekstraksi fitur selanjutnya karena gambar dengan komponen warna *grayscale* (1 *byte*) lebih sedikit jumlahnya dibandingkan gambar dengan komponen *RBG* (3 *byte*). Salah satu teknik yang dapat digunakan adalah dengan menggunakan rumus (Kumar & Verma, 2010) :

$$L_y = 0.333F_R + 0.5F_G + 0.1666F_B \quad (1)$$

- Proses segmentasi gambar
  Proses ini bertujuan untuk memisahkan obyek *(foreground)* dengan latar belakang gambar *(background)* agar obyek lebih mudah diproses untuk proses selanjutnya. Proses ini biasanya dilakukan pada gambar *grayscale* dan proses ini menghasilkan gambar *binary* (1 *bit*). Salah satu teknik yang dapat digunakan adalah dengan menggunakan nilai ambang batas *(threshold)*. Jika nilai sebuah piksel adalah lebih kecil dari nilai ambang batas, maka dianggap 0 (dianggap sebagai *background*). Jika nilai sebuah piksel adalah lebih besar sama dengan dari nilai ambang batas, maka dianggap 1 (dianggap sebagai *foreground*)
- Proses dilasi
  Proses ini bertujuan untuk menutup lubang-lubang kecil *(hole)* yang ada pada gambar atau menggabungkan daerah-daerah penting *(regions of interest)* pada gambar yang perbedaannya kecil menjadi sebuah daerah yang lebih besar sekaligus memperbesar ukuran daerah penting. Proses ini biasanya dilakukan pada gambar *binary* (1 *bit*). Proses ini biasanya dilakukan sebelum proses deteksi garis tepi gambar agar hasil deteksi garis tepi menjadi lebih akurat. Proses ini biasanya dilakukan pada gambar *binary*.
- Proses erosi
  Proses ini bertujuan untuk menghilangkan detail-detail kecil yang ada pada gambar sekaligus mengurangi ukuran daerah penting *(regions of interest)*. Proses ini biasanya dilakukan pada gambar *binary* (1 *bit*). Proses ini biasanya dilakukan sebelum proses deteksi garis tepi gambar agar hasil deteksi garis tepi menjadi lebih akurat. Proses ini biasanya dilakukan pada gambar *binary*.
- Proses deteksi garis tepi

Proses ini bertujuan untuk mendapatkan garis tepi dari sebuah obyek sehingga dapat digunakan untuk menganalisa obyek dalam proses pengenalan obyek. Proses ini biasanya dilakukan pada gambar *binary.* Salah satu algoritma yang sering digunakan dalam mendeteksi garis tepi adalah algoritma *Canny*

Proses ekstraksi fitur bentuk obyek terdiri dari beberapa langkah yaitu penghitungan *circularity*, penghitungan *rectangularity* (Paul, 2013).

- Proses penghitungan *circularity*
  Circularity menunjukkan sejauh mana sebuah obyek tersebut bulat. Nilai *circularity* berkisar antara 0 hingga 1. Semakin dekat dengan angka 1 berarti obyek tersebut semakin bulat. Semakin dekat dengan angka 0 berarti obyek tersebut semakin tidak bulat.

$$C = \frac{K^2}{L.4\pi} \tag{2}$$

- Proses penghitungan *rectangularity*
  Rectangularity merupakan rasio perbandingan antara luas obyek (A) dengan luas kotak pembatas minimal obyek (A_m). Kotak pembatas adalah sebuah kotak yang berisi obyek tersebut dimana semua sisi dari obyek tersebut menyentuh obyek tersebut. Nilai *rectangularity* berkisar antara 0 hingga 1. Semakin dekat dengan angka 1 berarti obyek tersebut semakin bulat. Semakin dekat dengan angka 0 berarti obyek tersebut semakin tidak kotak.

$$R = \frac{A}{A_m} \tag{3}$$

Proses ekstraksi fitur tekstur obyek terdiri dari beberapa langkah yaitu penghitungan varian, penghitungan *skewness*, penghitungan kurtosis, penghitungan *relative smoothness.* Untuk melakukan penghitungan statistik ordo pertama, diperlukan angkan *mean* dimana didapatkan dari rumus :

$$\mu = \frac{\sum_{i=0}^{n} X_i}{n} \tag{4}$$

dimana n adalah jumlah tingkat abu-abu dan Xi adalah jumlah piksel pada tingkat abu-abu ke i.

- Penghitungan varian
  Untuk menghitung varian, menggunakan rumus :

$$\sigma^2 = \sum_n (f_n - \mu)^2 P(f_n) \tag{5}$$

dimana $P(f_n)$ adalah probabilitas kemunculan intensitas pada gambar. Varian ini menunjukkan variasi elemen dari kurva histogram.

- Penghitungan *skewness*
  Untuk menghitung *skewness*, menggunakan rumus :

$$\alpha^3 = \frac{1}{\sigma^3} \sum_n (f_n - \mu)^3 P(f_n) \tag{6}$$

Skewness menunjukkan inklaniasi relatif pada kurva histogram gambar

- Penghitungan kurtosis
  Untuk menghitung kurtosis, menggunakan rumus :

$$\alpha^4 = \frac{1}{\sigma^4} \sum_n (f_n - \mu)^4 P(f_n) - 3 \tag{7}$$

Kurtosis tingkat tertinggi pada kurva histogram gambar

- Penghitungan *relative smoothness*
  Untuk menghitung *relative smoothness*, menggunakan rumus :

$$R = 1 - \left(\frac{1}{1+\sigma^2}\right) \tag{8}$$

Relative smoothness menunjukkan level *relative smoothness* pada sebuah bentuk gambar.

2.2.1. OpenCV

OpenCV (Open Source Computer Vision) merupakan library yang dibuat oleh Intel untuk menyelesaikan permasalahan visi komputer dan bersifat *open-source*. *OpenCV* ditulis di dalam bahasa C dan C++ dan dapat dijalankan di sistem operasi *Linux*, *Windows* dan *Mac OS X*. Saat ini sedang dilakukan pengembangan agar bisa digunakan pada bahasa *Phyton*, *Ruby*, *Matlab* dan bahasa pemrograman lain (Bradski, & Kaehler, 2008).

OpenCV didesain untuk efisiensi komputasi dengan difokuskan pada penerapan visi komputer yang *real-time*. Salah satu tujuan dari *OpenCV* adalah menyediakan sebuah infrastruktur visi komputer yang mudah digunakan yang dapat membantu banyak orang dalam membangun aplikasi computer vision dengan cepat (Bradski, & Kaehler, 2008).

OpenCV merupakan sebuah library tingkat tinggi yang mengimplementasikan algoritma-algoritma seperti *calibration techniques (camera calibration), feature detection (feature) and tracking (optical flow), shape analysis (geometry, contour processing), motion analysis*

(motion templates, estimators), 3D reconstruction (view morphing), object segmentation and recognition (histogram, embedded hidden markov models, eigen objects). Beberapa contoh area penerapannya adalah *human-computer interaction, object indentification, segmentation and recognition, face recognition, gesture recognition, motion tracking, ego motion, motion understanding, structure from motion (SFM),* dan *mobile robotics* (Intel, 2000).

Struktur *OpenCV* dibagi menjadi empat komponen utama yaitu *CV*, *MLL*, *HighGUI*, dan *CXCORE* (Bradski, & Kaehler, 2008). Komponen *CV* terdiri dari pengolahan citra dasar dan algoritma visi komputer pada tingkat high-level. Komponen *ML* adalah komponen *Machine Learning* dimana berisi banyak alat klasifikasi (*classifier*) secara statistik dan pengelompokan (*clustering*). Komponen *HighGUI* berisi *I/O routine* dan fungsi-fungsi untuk menyimpan dan memuat gambar dan video. Komponen *CXCORE* berisi struktur data dasar dan konten.

3. JARINGAN SYARAF TIRUAN

Jaringan syaraf tiruan adalah sebuah sistem yang memproses informasi yang memiliki karakter performa yang sama dengan jaringan syaraf biologis. Jaringan syaraf tiruan adalah sebuah sistem yang bersifat adaptif yang dapat belajar untuk melakukan sebuah fungsi (pemetaan data masukan dan data keluaran) berdasarkan data (Fausett, 1993). Adaptif berarti parameter-parameter yang dimiliki sistem dapat berubah (disebut fase pelatihan). Setelah fase pelatihan selesai dimana parameter-parameter yang dimiliki sistem sudah ditentukan, maka sistem dapat digunakan untuk menyelesaikan permasalahan yang ada (disebut fase uji coba).

Sebuah jaringan syaraf tiruan terdiri dari sejumlah elemen pemroses informasi yang disebut neuron atau node. Sebuah neuron terhubung dengan neuron yang lain melalui penghubung koneksi berarah yang diserta sebuah bobot. Sebuah bobot mewakili informasi yang sedang digunakan oleh jaringan tersebut dalam menyelesaikan sebuah permasalahan, salah satunya adalah pengenalan pola. Sebuah neuron memiliki sebuah fungsi aktivasi yang akan memproses sinyal masukan dan mengirimkan sinyal keluaran kepada neuron-neuron yang lainnya.

Secara umum, jaringan syaraf tiruan dikembangkan sesuai cara kerja jaringan syaraf biologis, dengan asumsi sebagai berikut :

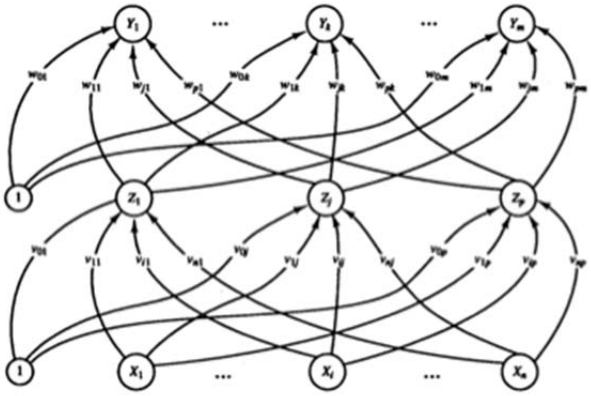
- Pemrosesan informasi muncul pada elemen-elemen yang sederhana, disebut neuron-neuron
- Sinyal disalurkan ke neuron-neuron tersebut melalui penghubung-penghubung koneksi
- Setiap penghubung koneksi memiliki sebuah bobot
- Setiap neuron mengaplikasikan sebuah fungsi aktivasi (biasanya non linier) pada sinyal masukan (jumlah dari sinyal masukan beserta bobotnya) untuk menentukan sinyal keluaran yang nantinya akan disalurkan ke semua neuron di lapisan selanjutnya sebagai sinyal masukan.

Jaringan syaraf tiruan dapat digolongkan menjadi dua macam, yaitu *single-layer network* (jaringan dengan satu lapisan) dan *multi-layer network* (jaringan dengan banyak lapisan). *Single-layer network* adalah jaringan syaraf tiruan yang hanya mempunyai sebuah lapisan penghubung (lapisan bobot). Jaringan syaraf tiruan ini memiliki lapisan masukan (*input layer*) dan lapisan keluaran (*output layer*). *Multi-layer network* adalah jaringan syaraf tiruan yang hanya mempunyai dua lapisan penghubung (lapisan bobot) atau lebih. Jaringan syaraf tiruan ini memiliki lapisan masukan (*input layer*), lapisan tersembunyi (*hidden layer*), dan lapisan keluaran (*output layer*). Neuron-neuron pada sebuah lapisan itu sepenuhnya terhubung dengan neuron-neuron pada lapisan selanjutnya tetapi tidak terhubung dengan neuron-neuron lain pada lapisan sebelumnya. *Multi-layer network* dapat menyelesaikan permasalahan yang lebih kompleks dibandingkan apa yang dapat *single-layer network* lakukan, tetapi proses pelatihannya lebih rumit.

3.1. Jaringan Syaraf Tiruan Dengan Banyak Lapisan

Jaringan syaraf tiruan ini terdiri dari sebuah lapisan masukan (input layer), satu atau lebih lapisan tersembunyi (hidden layer) dan sebuah lapisan keluaran (output layer). Lapisan masukan terdiri dari n neuron. Lapisan tersembunyi terdiri dari p neuron. Lapisan keluaran terdiri dari m neuron. Penggunaan lebih dari satu lapisan tersembunyi memang terkadang menguntungkan dalam beberapa kasus, tetapi menggunakan satu lapisan tersembunyi saja sudah cukup. Setiap neuron pada lapisan tersembunyi dan lapisan keluaran mempunyai sebuah bias. Bias ini sama seperti nilai bobot pada koneksi penghubung dimana nilainya selalu 1.

Di bawah ini adalah contoh sebuah jaringan syaraf tiruan dengan sebuah lapisan tersembunyi. Neuron-neuron pada lapisan masukan diberi nama X1 hingga Xn. Neuron-neuron pada lapisan tersembunyi diberi nama Z1 hingga Zp. Neuron-neuron pada lapisan keluaran diberi nama Y1 hingga Ym. Koneksi penghubung antara neuron di lapisan masukan dengan neuron di lapisan tersembunyi diberi nama vab dimana a berarti urutan neuron asal di lapisan masukan dan b berarti urutan neuron tujuan di lapisan tersembunyi. Koneksi penghubung antara neuron di lapisan tersembunyi dengan neuron di lapisan keluaran diberi nama wab dimana a berarti urutan neuron asal di lapisan tersembunyi dan b berarti urutan neuron tujuan di lapisan keluaran. Koneksi penghubung antara bias di lapisan masukan dengan neuron di lapisan tersembunyi diberi nama v0b dimana b berarti urutan neuron tujuan di lapisan tersembunyi. Koneksi penghubung antara bias di lapisan tersembunyi dengan neuron di lapisan keluaran diberi nama w0b dimana b berarti urutan neuron tujuan di lapisan keluaran.



Gambar 1. Jaringan Syaraf Tiruan Dengan Banyak Lapisan.

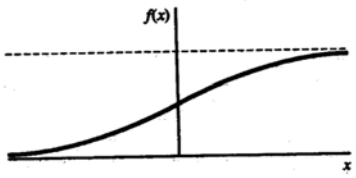
Simbol dan istilah yang sering digunakan dalam jaringan syaraf tiruan dengan banyak lapisan dapat dilihat pada Tabel 1.

Tabel 1. Simbol dan Istilah Pada Jaringan Syaraf Tiruan Dengan Banyak Lapisan

Simbol	Penjelasan
x	Vektor masukan (digunakan pada fase pelatihan) **x = (x_1, … , x_n)**
t	Vektor keluaran yang merupakan target yang diharapkan (digunakan pada fase pelatihan) t = (t_1, … , t_m)
δ_k	Nilai koreksi kesalahan (*error*) untuk penyesuaian nilai bobot w_{jk} pada neuron Y_k pada lapisan keluaran. Informasi ini yang dirambatkan terbalik ke neuron-neuron yang berhubungan pada lapisan tersembunyi
δ_j	Nilai koreksi kesalahan (*error*) untuk penyesuaian nilai bobot v_{ij} dirambatkan terbalik dari lapisan keluaran menuju neuron-neuron yang berhubungan pada lapisan tersembunyi
α	Laju pembelajaran (*learning rate*)
$\mathbf{X_i}$	Neuron ke – i pada lapisan masukan
$\mathbf{Y_k}$	Neuron ke – k pada lapisan keluaran
$\mathbf{Z_j}$	Neuron ke – j pada lapisan tersembunyi
$\mathbf{v_{0j}}$	Nilai bias pada neuron ke – j pada lapisan tersembunyi
$\mathbf{w_{0k}}$	Nilai bias pada neuron ke – k pada lapisan keluaran

Fungsi aktivasi yang dapat digunakan pada jaringan syaraf tiruan dengan banyak lapisan harus mempunyai sifat kontinu, dapat dibedakan (*differentiable*), dan tidak turun secara monoton (*monotonically non-decreasing*) (Shenouda, 2006). Dari sejumlah fungsi aktivasi yang ada, fungsi sigmoid biner (*binary sigmoid*) dan fungsi sigmoid bipolar (*bipolar sigmoid*). Berdasarkan fungsi aktivasi yang digunakan, sinyal masukan dan sinyal keluaran harus ditransformasi terlebih dahulu sehingga memiliki pola sinyal masukan dan sinyal keluaran yang sama. Fungsi sigmoid biner mempunyai jangkauan nilai 0 dan 1 dimana persamaannya dapat dilihat pada Persamaan 9 dan dapat diilustrasikan pada Gambar 2.

$$f_1(x) = \frac{1}{1+\exp(-x)'} \qquad (1)$$
$$f_1'(x) = f_1(x)[1 - f_1(x)]$$



Gambar 2. Jangkauan Fungsi Sigmoid Biner

3.2. Jaringan Syaraf Tiruan Untuk Pengenalan Tas Tangan

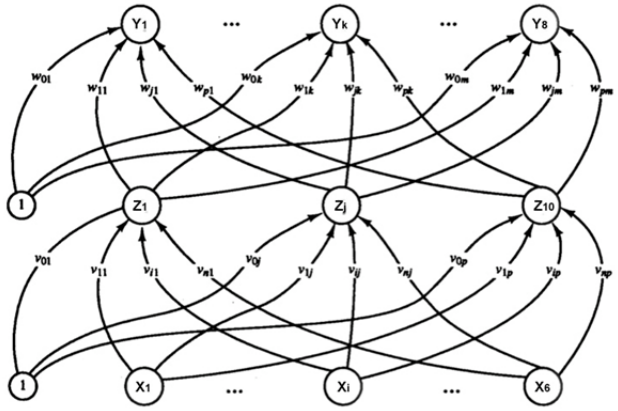
Jaringan syaraf tiruan yang digunakan dalam mengenali tas tangan adalah jaringan syaraf tiruan dengan banyak lapisan. Jaringan syaraf tiruan dengan banyak lapisan terdiri dari tiga lapisan yaitu lapisan masukan, lapisan tersembunyi dan lapisan keluaran.

Lapisan masukan terdiri dari 6 neuron dimana 2 neuron untuk fitur geometri (X1, X2) dan 4 neuron untuk fitur tekstur (X3, X4, X5, X6). Data untuk 6 neuron tersebut merupakan hasil ekstraksi fitur pada data masukan.

Lapisan tersembunyi terdiri dari 10 neuron (Z1, Z2, Z3, …, Z8, Z9, Z10) dimana jumlah tersebut didapatkan sesuai dengan rumus pada bagian sebelumnya yaitu dua per tiga dari jumlah neuron pada lapisan masukan ditambah jumlah neuron pada lapisan keluaran. Fungsi aktivasi yang digunakan pada lapisan ini adalah fungsi sigmoid.

Lapisan keluaran terdiri dari 8 neuron (Y1, Y2, Y3, …, Y6, Y7, Y8). Jumlah neuron pada lapisan ini didapatkan dari jumlah tipe tas tangan yang dapat dikenali oleh jaringan syaraf tiruan ini yaitu ada 8 tipe tas tangan dimana setiap neuron merepresentasikan sebuah tipe tas tertentu. Fungsi aktivasi yang digunakan lapisan ini adalah fungsi sigmoid. Nilai keluaran yang dihasilkan dari fungsi aktivasi tersebut adalah sebuah nilai desimal berkisar 0 hingga 1. Nilai keluaran tersebut selanjutnya diberi ambang batas pada nilai 0,8 agar menghasilkan nilai akhir 0 dan 1. Jika nilai keluaran >= 0,8 maka nilai akhirnya adalah 1. Jika nilai keluaran < 0,8 maka nilai akhirnya adalah 0. Neuron yang menghasilkan nilai akhir 1 merupakan neuron yang terpilih sebagai hasil dari klasifikasi.

Pada jaringan syaraf tiruan ini, setiap neuron pada lapisan tersembunyi dan lapisan keluaran mempunyai sebuah bias. Bias ini sama seperti nilai bobot pada koneksi penghubung dimana nilainya selalu 1.



Gambar 3. Desain Jaringan Syaraf Tiruan Dengan Banyak Lapisan Untuk Mengenali Tas Tangan

3.3. Pelatihan Jaringan Syaraf Tiruan Dengan Perambatan Balik

Jaringan syaraf tiruan yang telah dibuat harus diberikan pelatihan agar dapat menjadi jaringan syaraf tiruan yang memiliki kepintaran dalam mengklasifikasikan data masukan baru. Kepintaran jaringan syaraf tiruan berada pada nilai bobot pada setiap koneksi penghubung antar neuron di semua lapisan. Jaringan syaraf tiruan yang telah dilatih akan menghasilkan perubahan nilai bobot pada setiap koneksi penghubung antar neuron di semua lapisan.

Beberapa parameter yang mempengaruhi tingkat keberhasilan tahap pelatihan pada jaringan syaraf tiruan adalah inisialisasi bobot, laju pelatihan, momentum, jumlah lapisan tersembunyi, lama pelatihan dan jumlah data pelatihan.

Algoritma perambatan balik *(backpropagation)* adalah salah satu algoritma yang dapat digunakan untuk melatih sebuah jaringan syaraf tiruan banyak lapisan dimana algoritma ini akan membandingkan nilai keluaran yang dihasilkan oleh jaringan syaraf tiruan *(actual output)* dengan nilai keluaran yang seharusnya *(expected output)* untuk menghasilkan *Mean Squared Error (MSE)*. Nilai MSE akan digunakan untuk memperbaiki nilai bobot pada setiap koneksi penghubung.

Algoritma ini terdiri dari tiga fase yaitu fase perambatan maju (*feedforward*) terhadap data masukan, fase perambatan terbalik terhadap kesalahan yang dihasilkan, dan fase penyesuaian kembali nilai bobot dan bias. Nilai bobot diisi secara acak pada waktu inisialisasi jaringan. Algoritma pelatihan ini merupakan *supervised training* karena disediakan sejumlah vektor masukan atau pola masukan dimana setiap masukan itu disertai juga dengan vektor keluaran atau pola keluaran.

Pada tahap perambatan maju, setiap neuron di lapisan masukan (Xi) menerima sinyal dan meneruskan sinyal tersebut ke semua neuron di lapisan tersembunyi (Z1, … , Zp). Kemudian, setiap neuron di lapisan tersembunyi itu (Zi) menghitung sinyal keluaran atau respon ke setiap neuron di lapisan keluaran (Y1, … , Ym) dengan menggunakan fungsi aktivasi yang telah ditentukan. Kemudian, setiap neuron di lapisan keluaran itu (Yi) menghitung sinyal keluaran atau respon terhadap sinyal masukan yang diberikan ke jaringan tersebut dengan menggunakan fungsi aktivasi yang telah ditentukan.

Pada tahap perambatan balik, setiap neuron pada lapisan keluaran (Yi) membandingkan sinyal keluaran yang dihasilkan itu (nilai aktual) dengan nilai keluaran yang seharusnya (nilai yang diharapkan) untuk menentukan nilai kesalahan (*error delta*). Nilai keluaran yang seharusnya itu didapatkan dari data pelatihan dimana data pelatihan terdiri dari sejumlah pasangan nilai masukan dan nilai keluaran. Berdasarkan nilai kesalahan tersebut, dapat dihitung nilai faktor δk (k = 1, … , m) yang digunakan untuk mendistribusikan terbalik nilai kesalahan pada neuron tersebut di lapisan keluaran (Yk) ke semua neuron di lapisan sebelumnya (lapisan tersembunyi) yang terhubung dengannya. Nantinya, nilai faktor δk juga digunakan untuk menyesuaikan nilai bobot pada koneksi penghubung dan nilai bias antara lapisan tersembunyi dengan lapisan keluaran. Dengan cara yang sama pula, dapat dihitung nilai faktor δj (j = 1, … , p) yang digunakan untuk mendistribusikan terbalik nilai kesalahan pada neuron tersebut di lapisan tersembunyi (Zj) ke semua neuron di lapisan sebelumnya (lapisan masukan) yang terhubung dengannya. Memang tidak begitu penting untuk mendistribusikan terbalik nilai kesalahan ke semua neuron di lapisan masukan, tetapi faktor δj digunakan untuk menyesuaikan nilai bobot pada koneksi penghubung dan nilai bias antara lapisan tersembunyi dengan lapisan keluaran.

Pada tahap penyesuaian kembali nilai bobot dan bias, semua nilai bobot pada koneksi penghubung dihitung lagi sesuai dengan nilai faktor δ yang didapatkan secara serentak. Nilai bobot pada koneksi penghubung dan bias pada setiap neuron di lapisan tersembunyi menuju lapisan keluaran didasarkan pada nilai faktor δk dan sinyal keluaran yang dihasilkan dari fungsi aktivasi. Nilai bobot pada koneksi penghubung dan bias pada setiap neuron di lapisan masukan menuju lapisan tersembunyi didasarkan pada nilai faktor δj dan sinyal keluaran yang dihasilkan dari fungsi aktivasi.

Untuk lebih lengkapnya, berikut adalah langkah-langkah pada algoritma pelatihan perambatan balik :

- Langkah 0 :
  Inisialisasi nilai bobot pada semua koneksi penghubung baik dari lapisan masukan menuju lapisan tersembunyi maupun dari lapisan tersembunyi menuju lapisan keluaran. Isi dengan angka acak yang kecil.
- Langkah 1 :
  Selama kondisi berhenti masih false, maka lakukan langkah 2 – langkah 9.
  - Langkah 2 :
    Untuk setiap pasangan data pelatihan (data masukan dan data keluaran), lakukan langkah 3 – 8.

    Tahap perambatan maju :
    - Langkah 3 :

Setiap neuron pada lapisan masukan (Xi, dimana i = 1, … , n) menerima sinyal (xi) dan meneruskan sinyal tersebut ke semua neuron di lapisan tersembunyi (Z1, … , Zp).

- Langkah 4 :
  Pada setiap neuron di lapisan tersembunyi itu (Zj, dimana j = 1, … , p), hitung nilai sinyal masukan dengan cara menjumlahkan semua sinyal masukan yang berbobot (menggunakan rumus yang ada di Gambar 2.27). Dari sinyal masukan yang didapatkan, gunakan fungsi aktivasi yang telah ditentukan untuk menghitung sinyal keluaran (menggunakan rumus yang ada di Gambar 2.28). Kemudian, kirimkan sinyal keluaran ini ke setiap neuron di lapisan keluaran (Y1, … , Ym).

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \tag{9}$$

$$z_j = f\left(z\_in_j\right) \tag{10}$$

  Catatan :
  Langkah ini dilakukan sebanyak jumlah lapisan tersembunyi.
- Langkah 5 :

Pada setiap neuron di lapisan keluaran itu (Yk, dimana k = 1, … , m), hitung nilai sinyal masukan dengan cara menjumlahkan semua sinyal masukan yang berbobot (menggunakan rumus yang ada di Gambar 2.29). Dari sinyal masukan yang didapatkan, gunakan fungsi aktivasi yang telah ditentukan untuk menghitung sinyal keluaran (menggunakan rumus yang ada di Gambar 2.30).

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk} \tag{11}$$

$$y_k = f\left(y\_in_k\right) \tag{12}$$

Tahap perambatan balik :

- Langkah 6 :
  Pada setiap neuron pada lapisan keluaran (Yk, dimana k = 1, … , m), didapatkan nilai kesalahan (error delta) antara nilai keluaran yang dihasilkan dengan nilai keluaran yang seharusnya. Hitung nilai faktor δk (menggunakan rumus yang ada di Gambar 2.31), delta nilai bobot pada koneksi penghubung (menggunakan rumus yang ada di Gambar 2.32), dan delta nilai bias (menggunakan rumus yang ada di Gambar 2.33). Kemudian, distribusikan terbalik nilai faktor δk pada neuron tersebut di lapisan keluaran (Yk) ke semua neuron di lapisan sebelumnya (lapisan tersembunyi) yang terhubung dengannya.

$$\delta_k = \left(t_k - y_k\right) f'(y\_in_k) \tag{13}$$

$$\Delta w_{jk} = \alpha \delta_k z_j \tag{14}$$

$$\Delta w_{0k} = \alpha \delta_k \tag{15}$$

  Untuk memudahkan pemahaman, nilai faktor δ yang didistribusikan terbalik itu diperlakukan seperti sinyal masukan pada jaringan syaraf tiruan ini tetapi dalam arah terbalik dimana lapisan keluaran dianggap sebagai lapisan masukan dan lapisan masukan dianggap sebagai lapisan keluaran. Itulah sebabnya, fase ini disebut backpropagation (perambatan terbalik).

  Catatan :
  Langkah ini juga dilakukan sebanyak jumlah lapisan tersembunyi, yaitu dari suatu lapisan tersembunyi ke lapisan tersembunyi sebelumnya.
- Langkah 7 :
  Pada setiap neuron di lapisan tersembunyi itu (Zj, dimana j = 1, … , p), hitung nilai masukan delta dengan cara menjumlahkan semua nilai masukan delta (menggunakan rumus yang ada di Gambar 2.34). Dari nilai masukan delta yang didapatkan, kalikan dengan turunan dari fungsi aktivasi yang telah ditentukan untuk menghitung nilai faktor δj (menggunakan rumus yang ada di Gambar 2.35). Hitung juga delta nilai bobot pada koneksi penghubung (menggunakan rumus yang ada di Gambar 2.36), dan delta nilai bias (menggunakan rumus yang ada di Gambar 2.37).

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \tag{16}$$

$$\delta_j = \delta\_in_j f'\left(z\_in_j\right) \tag{17}$$

$$\Delta v_{ij} = \alpha \delta_j x_i \tag{18}$$

$$\Delta v_{0j} = \alpha \delta_j \tag{19}$$

Tahap penyesuaian kembali nilai bobot dan bias :

- Langkah 8 :
  Pada setiap neuron di lapisan keluaran (Yk, dimana k = 1, … , m), ubah nilai bobot pada koneksi penghubung dan nilai bias yang menghubungkan antara lapisan tersembunyi dengan lapisan keluaran.

  Pada setiap neuron di lapisan tersembunyi (Zj, dimana j = 1, … , p), ubah nilai bobot pada koneksi penghubung dan nilai bias yang menghubungkan antara lapisan masukan dengan lapisan tersembunyi.
- Langkah 9 :
  Periksa kondisi berhenti. Pada umumnya, kondisi berhenti yang dipakai adalah jumlah epoch (adalah sebuah iterasi atau pengulangan penuh pada seluruh data pelatihan) atau nilai MSE (Mean Square Error). Pelatihan jaringan akan dihentikan jika jumlah epoch saat itu sudah melebihi jumlah maksimum jumlah epoch yang ditentukan atau jika nilai MSE saat itu lebih kecil dari nilai MSE yang ditentukan.

Algoritma pelatihan jaringan syaraf tiruan perambatan balik digunakan untuk melatih jaringan syaraf tiruan untuk pengenalan tipe tas. Inisialisasi bobot pada koneksi penghubung menggunakan metode *Nguyen – Widrow*. Pelatihan jaringan syaraf tiruan menggunakan nilai laju pelatihan sebesar 0,1 dan momentum sebesar 0,8 dimana penentuan laju pelatihan dan momentum dilakukan secara *trial-and-error*. Jumlah data pelatihan yang digunakan adalah 414 data dimana berisi data pelatihan untuk semua kelas. Lama pelatihan jaringan syaraf tiruan ditentukan oleh dua parameter yaitu jumlah *epoch* dan *Mean Squared Error (MSE)*. Pelatihan jaringan syaraf tiruan akan berhenti apabila salah satu parameter tersebut terpenuhi. Jumlah *epoch* yang digunakan adalah 1.000 *epoch*. Nilai *Mean Squared Error (MSE)* yang digunakan adalah 0.02.

Tingkat akurasi klasifikasi yang dihasilkan pada jaringan syaraf tiruan ini adalah 71.1% dimana pelatihan berhenti pada *epoch* ke 980 dengan nilai *MSE (Mean Squared Error)* sebesar 0.019641. Nilai bobot yang didapatkan dari tahap pelatihan ini akan digunakan pada tahap uji coba.

4. UJI COBA DAN DISKUSI

Jaringan syaraf tiruan yang telah dilatih harus diuji coba untuk mengetahui seberapa akurat jaringan syaraf tiruan tersebut dalam mengklasifikasikan data masukan baru. Sebuah jaringan syaraf tiruan dikatakan memiliki kepintaran yang cukup dalam mengklasifikasi data masukan baru jika tingkat akurasinya di bawah *Minimum Squared Error (MSE)* yang ditetapkan.

Jumlah data uji coba yang digunakan adalah 28 data dimana berisi data pelatihan untuk semua kelas. Berdasarkan hasil uji coba yang dilakukan, tingkat akurasi yang dihasilkan adalah 78,57%. Sebanyak 21 data berhasil diklasifikasikan dengan benar dan 7 data tidak berhasil diklasifikasikan dengan benar.

Berdasarkan tabel 2, jaringan syaraf tiruan mampu mengenali tipe tas 1, tipe tas 2, tipe tas 5 dan tipe tas 8 dengan sangat baik (100%). Jaringan syaraf tiruan mampu mengenali tipe tas 3 dengan baik (75%). Jaringan syaraf tiruan mampu mengenali tipe tas 4 dan tipe tas 7 dengan cukup baik (50%). Namun, jaringan syaraf tiruan tidak mampu mengenali tipe tas 6 (0%). Hal ini disebabkan karena jumlah data pelatihan untuk kelas 6 hanya 1 buah.

Tabel 2. Tabel hasil uji coba jaringan syaraf tiruan

Tipe Tas	Prosentasi Jumlah Data Pelatihan	Jumlah dikenali	Jumlah tidak dikenali	Tingkat akurasi
Tipe Tas 1	14,28%	4	0	100%
Tipe Tas 2	14,28%	4	0	100%
Tipe Tas 3	14,28%	3	1	75%
Tipe Tas 4	14,28%	2	2	50%
Tipe Tas 5	10,73%	3	0	100%
Tipe Tas 6	3,59%	0	1	0%
Tipe Tas 7	14,28%	2	2	50%
Tipe Tas 8	14,28%	4	0	100%
	100%	78,57%	21,43%	

Penggunaan jumlah data pelatihan yang cukup dan merata untuk setiap kelas dapat meningkatkan tingkat akurasi jaringan syaraf tiruan tersebut.

5. KESIMPULAN DAN SARAN

Pada akhir penelitian, terdapat beberapa kesimpulan yang dapat ditarik yaitu :

- Fitur geometri dan fitur tekstur yang digunakan sudah dapat mengenali obyek tas tangan pada latar belakang sederhana.
- Jaringan syaraf tiruan perambatan balik yang dibangun dapat mengklasifikasikan data masukan dengan tingkat akurasi 78,57%.

Berdasarkan hasil penelitian tersebut, terdapat beberapa saran bagi perbaikan penelitian ini, yaitu :

- Penambahan fitur lain selain fitur tekstur dan geometri dapat memberikan kontribusi dalam meningkatkan tingkat akurasi pada jaringan syaraf tiruan.
- Penggunaan algoritma lain dalam ekstraksi fitur untuk menghasilkan fitur yang lebih baik.
- Meningkatkan jumlah data pelatihan untuk meningkatkan tingkat akurasi pada jaringan syaraf tiruan untuk semua kelas secara merata.
- Perbaikan struktur jaringan syaraf tiruan pada neuron output agar lebih efisien.
- Penggabungan teknik klasifikasi dan segmentasi dapat meningkatkan tingkat akurasi karena tingkat persamaan tas tangan relatif tinggi antar tiap merek.

DAFTAR PUSTAKA

Bradski, G., & Kaehler, A. (2008). Learning Open CV (1st ed.). Sebastopol, CA: O'Reilly Media, Inc.

Fausett, L. (1993). Fundamentals of Neuron Network (1st ed.). USA: Pearson.

Intel. (2000). Open Source Computer Vision Library Reference Manual. USA: Intel Corporation.

Kumar, T., & Verma, K. (2010). A Theory Based on Conversion of RGB Image to Gray Image. International Journal of Computer Application, 7(2).

Mingqiang, Y., Kidiyo, K., & Josepj, R. (2008). A Survey of Shape Feature Extraction Techniques.

Paul, L. R. (2013). Measuring Shape: Ellipticity, Rectangularity and Triangularity. United Kingdom: Department of Computer Science, Cardiff Universty.

Shenouda, E. A. (2006). A Quantitative Comparison of Different MLP Activation Functions in Classification. doi:10.1007/11759966_125.