JURNAL TEKNOLOGI INFORMASI
PROGRAM STUDI TEKNIK INFORMATIKA DAN SISTEM INFORMASI, UNIVERSITAS BUNDA MULIA
Volume 11, Nomor 2, Agustus 2015
ISSN: 1979-1496

Perbandingan Hebb-Rule Dan Perceptron Dalam Segmentasi Citra Menggunakan *Input* Variasi RGB

Teady Matius Surya Mulyana
tmulyana@bundamulia.ac.id, teadymatius@yahoo.com
Teknik Informatika Universitas Bunda Mulia

ABSTRAK

Algoritma Hebb-Rule merupakan algoritma pembelajaran pada jaringan saraf tiruan. Algoritma ini dapat mencari nilai bobot peran dari masing-masing input. Sehingga dapat menghasilkan output yang terpisah secara linear sesuai dengan kondisi yang diberikan ketika dilakukan pelatihan terhadap sistem.
Variasi channel RGB yang dijadikan tujuh variasi, yaitu R, G, B, RG, RB, GB dan RGB. Ketujuh variasi tersebut dapat di threshold nilai grayscale nya, sehingga menghasilkan citra binernya masing-masing. Ketujuh variasi channel ini akan menjadi input pada jaringan saraf tiruan. Perbandingan ketujuh variasi tersebut diimplementasikan pada perkalian masing-masing input dengan masing-masing bobotnya. Output yang dihasilkan diadaptasi hanya untuk 0 dan 1. Algoritma Hebb-rule dan algoritma dapat digunakan pada pelatihan untuk menghasilkan bobot yang akan menentukan peranan dari masing-masing input variasi channel RGB pada segmentasi citra untuk menentukan warna yang akan dianggap obyek dan warna yang dianggap latar belakang. Perbedaan dari kedua algoritma training ini adalah pada rumus pencarian bobot baru. Dengan perbedaan tersebut menghasilkan perbedaan bobot dan waktu iterasi yang dihasikan.

Kata Kunci: *Segmentasi Citra , Perceptron, Hebb-Rule, Variasi RGB*

PENDAHULUAN

Segmentasi dan Citra Biner

Pengenalan obyek pada citra harus didahului dengan memisahkan antara obyek dengan latar belakang sehingga mudah untuk dianalisa[12][6]. Pemisahan ini harus menghasilkan citra biner yang hanya mempunyai dua warna yaitu hitam dan putih. Obyek akan dihadirkan dengan warna hitam, sedangkan latar belakang akan dihadirkan dengan warna putih[12]. Proses binerisasi ini mempunyai banyak kesulitan untuk menentukan warna yang akan dianggap obyek dan warna yang dianggap latar belakang[12][6].

Mukherjee (Mukherjee, 2010) menuliskan bahwa metode paling sederhana untuk melakukan segmentasi citra adalah *tresshold*ing[12]. *Thresholding* adalah teknik yang sangat penting pada segmentasi citra, penanganan dan deteksi obyek. *Output* dari proses *thresholding* citra biner yang mempunyai nilai skala keabuan 0 (hitam) yang mengindikasikan piksel yang berisi obyek atau target yang akan dianalisa dan nilai skala keabuan 1 (putih) yang diindikasikan sebagai latar belakang[12]. Ridler (Ridler, 1978) menjelaskan bahwa algoritma binarisasi dalam hal ini adalah prosedur untuk melakukan *threshold* optimal nilai untuk setiap piksel [18].

Model Warna RGB

Salah satu model warna citra yang banyak dipergunakan dalam penelitian pengolahan citra digital adalah model warna RGB. Sutoyo (Sutoyo, 2009)[21] menjelaskan setiap piksel pada citra warna mewakili warna dasar yang merupakan kombinasi dari tiga warna dasar merah (*Red*), hijau (*Green*) dan biru (*Blue*), dimana setiap warna mempunyai gradasi sebanyak 256 warna dengan variasi intensitas cahaya antara 0 sampai 255. Model ini disebut model warna RGB. Variasi dari gabungan ketiga intensitas cahaya inilah yang akan menghasilkan variasi warna-warna yang berbeda-beda.

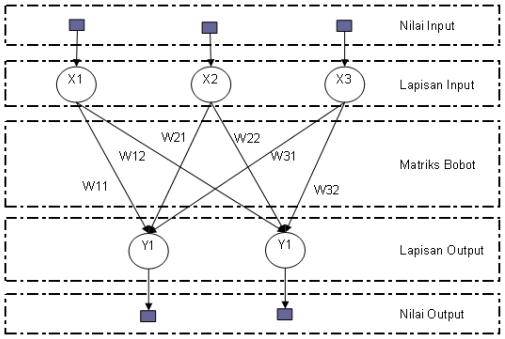
Model warna RGB dihasilkan dari tiga kombinasi warna utama (*Red*/merah, *Green*/hijau, dan *Blue*/biru) yang diturunkan menjadi nama model, dan pada *spectrum* cahaya yang dikombinasikan akan menghasilkan sebuah warna [9][22].

Citra Biner Pada Model Warna RGB

Pada model warna RGB, citra biner adalah citra yang berisi dua warna, yaitu hitam dan putih. Warna hitam dinotasikan dengan 0. Pada model RGB warna hitam dihasilkan dengan memberikan nilai 0 pada semua *channel* warna R, G dan B. Sedangkan warna putih yag bernilai biner 1, pada model RGB dihasilkan dengan memberikan nilai 255 pada semua *channel*, G dan B. semuanya dalam rentang nilai 0 sampai 255 [2][13].

Jaringan Syaraf Tiruan

Hermawan (Hermawan, 2006)[7] menjelaskan bahwa jaringan syaraf tiruan adalah sistem komputasi yang arsitekturnya diilhami dari cara kerja sel syaraf biologis otak manusia.



Gambar 1. Arsitektur jaringan syaraf tiruan dengan lapisan tunggal

Arsitektur jaringan syaraf tiruan dengan lapisan tunggal terdiri dari *input* matriks bobot, dan *output*. Seperti yang diperagakan pada gambar 1. Matriks bobot adalah suatu lapisan penentu *output* berdasarkan *input* yang ada. Untuk mendapatkan matriks bobot didapat dari inisialisasi bobot yang kemudian dilakukan pembelajaran sampai dihasilkan nilai bobot yang sesuai dan akan menghasilkan obyek sesuai dengan kriteria yang diharapkan.

Secara umum, rumus dari *output* Jaringan Saraf Tiruan adalah seperti yang ditunjukkan pada persamaan (1). Setiap set *input* i sampai n pada data ke j dikalikan dengan bobotnya hasilnya dijumlahkan sehingga menghasilkan sebuah *output* S dari kumpulan data j.

$$S_j = \sum_{i=0}^{n} a_i w_{ji} \quad \text{..... (1)}$$

keterangan:

- Sj adalah *output* dari sekumpulan data ke j
- ai adalah *input* data set ke i
- wji adalah bobot j dari data set j
- n adalah jumlah *input*

Nilai *output* Xj berkisar antara -1, 0 dan 1. Dihasilkan berdasarkan nilai Sj yang didapat. Jika Sj > 0 maka *output* Xj akan bernilai 1. Jika Sj = 0 maka *output* Xj akan bernilai 0. Jika Sj < 0 maka *output* Xj akan bernilai -1. *Output* Xj dapat juga dipolarkan menjadi -1 dan 1. Dengan menyesuaikan nilai S.

Perceptron

Khardon (Kahrdon, 2007)[10] dan Desiani (Desiani, 2006)[5], menjelaskan Perceptron merupakan algoritma *training* yang akan melakukan perbaikan nilai bobot sedemikian rupa sehingga didapat pengelompokan yang terpisah secara linear. Atau dengan kata lain terpisah dengan nilai yang berlawanan.

Pada setiap set *input*, semua *input* pada set tersebut akan dihitung *output* nya dengan persamaan (2). Persamaan tersebut hampir sama dengan persamaan (1) yang membedakan adalah adanya bias.

$$S_j = \sum_{i=0}^{n} a_i w_{ij} + bias \quad \text{..... (2)}$$

keterangan:

- Sj adalah *output* dari sekumpulan data ke j
- ai adalah *input* data set ke i
- wji adalah bobot j dari data set j
- n adalah jumlah *input*
- bias adalah nilai *input* bias.

Jika *output* X sesuai dengan kondisi contoh belajar, maka pembelajaran diteruskan pada contoh berikutnya.

Sedangkan jika tidak sama maka akan dilakukan penghitungna bobot sampai didapatkan *output* yang sesuai dengan contoh. Perbaikan bobot dilakukan berdasarkan bobot yang sebelumnya ditambhkan dengan kecepatan belajar dikalikan dengan selisih *output* dengan sampel, dan dikalikan dengan *input*. Seperti yang tercantum pada persamaan (3).

$$w_{ji} = w_{ji} + C(t_j - x_j) \times a_i \text{ ..... (3)}$$

Keterangan:
- w_{ji} : bobot ke i dari set data j
- C : kecepatan belajar
- t_j : *output* yang diharapkan dari set data j
- x_j : *output* aktual dari set data j
- a_i : *input* ke i

Sedangkan bias juga akan diperbaiki dengan bias baru berdasarkan bias sebelumya dengan persamaan (4).

$$bias_j = bias_j + C(t_j - x_j) \text{ . (4)}$$

Keterangan:
- $bias_j$: bias dari set data j
- C : kecepatan belajar
- t_j : *output* yang diharapkan dari set data j
- x_j : *output* aktual dari set data j

Salah satu contoh proses binarisasi citra dengan menggunakan perceptron dengan *input* yang berbeda adalah seperti yang dilakukan oleh Chen (Chen, 1992)[4]. Chen menggunakan rentang frekuensi dari nilai skala keabuan.

Hebb-Rule

Berbeda dengan perceptron yang akan melakukan proses perbaikan bobot yang rumit, hebb-Rule mempunyai alogritma *training* yang lebih sederhana. Hermawan (Hermawan, 2006)[7], Huyck, (Huyck, 2013)[8], Pugh, (Pugh, 2014)[17], Ahuja, (Ahuja,2013)[1], dan Sathasivam (Sathasivam, 2011)[19], menjelaskan Hebb rule merupakan algoritma *training* yang akan melakukan perbaikan nilai bobot sedemikian rupa sehingga jika ada dua neutron yang terhubung dan keduanya pada kondisi hidup pada saat yang sama, maka bobot keduanya akan dinaikkan. Data yang disajikan dalam bentuk bipolar. Bobot baru diperoleh dari penjumlahan bobot lama dengan aktivasi unit *input* dan unit *output* dengan persamaan (5).

$$w_i(\text{baru}) = w_i(\text{lama}) + x_i\, y \text{ ..... (5)}$$

keterangan:
- w_i : bobot ke i
- x_i : *input* ke i
- y : *output* yang diharapkan

Pencarian bobot pada hebb-rule lebih sederhana daripada perceptron, pada hebb-rule bobot baru untuk tiap-tiap *input* didapat dari bobot lama ditambah dengan *input* sampel ke i yang dikalikan dengan *output* yang diharapkan. Seperti yang dijelaskan oleh Huyck, (Huyck, 2013)[8], Pugh, (Pugh, 2014)[17], Ahuja, (Ahuja,2013)[1], dan Sathasivam (Sathasivam, 2011)[19].

Bias pada hebb-rule juga akan diperbaiki dengan bias baru berdasarkan bias sebelumya dengan persamaan (6).

$$bias_{jbaru} = bias_{jlama} + y \text{ ..... (6)}$$

keterangan:
- $bias_{jbaru}$: bias baru dari set data j
- $bias_{jlama}$: bias lama dari set data j
- y : *output* yang diharapkan

Bias pada hebb-rule dihitung dengan cara yang lebih sederhana daripada bias pada perceptron. Perceptron akan menambahkan bias lama dengan hasil kali kecepatan belajar dengan selisih *output* aktual dari *output* yang diharapkan. Sedangkan Hebb-rule hanya menambahkan dengan bias lama dengan *output* yang diharapkan.

Variasi channel RGB

Mulyana (Mulyana, 2013)[13], dalam penelitian mengenai *visible watermarking* menggunakan nilai *grayscale* menjelaskan Citra digital terbentuk dari piksel-piksel yang mempunyai intensitas warna merah, hijau dan biru yang dikomposisikan dengan berbagai variasi komposisi intensitas ketiganya, sehingga menghasilkan warna-warna tertentu pada posisi piksel tertentu, dan terkumpul pada

posisi tertentu sehingga terbentuk menjadi sebuah citra.

Jika pada ketiga *channel* RGB intensitas warna tersebut dinaikkan serentak ketiga-tiga nya dengan nilai yang sama, maka akan menghasilkan piksel yang lebih cerah dengan warna yang sama dengan piksel sebelumnya. Demikian juga Jika ketiga *channel* RGB intensitas warna tersebut diturunkan serentak ketiga-tiga nya dengan nilai yang sama, maka akan menghasilkan piksel yang lebih gelap dengan warna yang sama dengan piksel sebelumnya.

Variasi RGB untuk memisahkan antara obyek dengan *background* menjadi citra biner. *Channel* yang terpilih nantinya akan di*tresshold* untuk menentukan nilai-nilai yang akan dianggap 0 dan nilai-nilai yang dijadikan 255, pada citra biner dianggap sebagai 1[13].

Dengan variasi *channel* R/G/B akan didapatkan kombinasi pilihan *channel* yang terdiri dari salah satu *channel* saja, yaitu *channel* R saja, *channel* G saja dan *channel* B saja. Kemudian kombinasi dua *channel* yang terdiri dari kombinasi *channel* R dan G, kombinasi *channel* R dan B, kombinasi *channel* G dan B. Serta kombinasi ketiga *channel* R, G dan B. Dengan demikian ketiga *channel* tersebut jika dikombinasikan akan menghasilkan tujuh variasi *channel* RGB yaitu R, G, B, RG, RB, GB dan RGB.

Penghitungan nilai kombinasi dari skala keabuan untuk jumlah *channel* yang dipilih akan disesuaikan dengan jumlah *channel*. Dengan berdasarkan pada proses konversi ke skala keabuan pada ketiga *channel* R, G dan B seperti yang dijelaskan oleh Low, (Low, 1991)[10], yang dapat dilihat pada persamaan (7). Persamaan ini sama seperti yang peneliti lakukan, misalkan Nobuyuki (Nobuyuki 1979), Chan (Chan, 2005), Sauvolla (Sauvolla, 1999)[11][3][16][20].

$$f(x,y) = (R(x,y) + G(x,y) + B(x,y))/3 \quad (7)$$

keterangan:

- f(x,y) : piksel abu-abu pada sel(x,y)
- R(x,y) : intensitas *channel Red* pada sel(x,y)
- G(x,y) : intensitas *channel Green* pada sel(x,y)
- B(x,y) : intensitas *channel Blue* pada sel(x,y)

Karena pada pilihan *channel* ini dapat dipilih beberapa kemungkinan satu *channel* saja, kombinasi dua *channel* ataupun kombinasi tiga *channel*, secara umum persamaan tersebut dapat disusun dengan persamaan (8). Seperti yang sudah dilakukan pada penelitian Mulyana, (mulyana, 2013)[14] dan (mulyana, 2014)[15].

$$f(x,y) = \sum_{i=1}^{n} C_i(x,y) \; / \; n \quad \text{..... (8)}$$

keterangan:

- f(x,y) : piksel abu-abu pada sel(x, y)
- n : jumlah *channel* yang dipilih
- Ci(x,y) : Pilihan *channel* ke i pada sel(x,y)

Proses berikutnya adalah mendapatkan nilai biner dari piksel tersebut. Dengan ketentuan *tresshold* hasil dari penskala-keabuan dari persamaan (1) dilakukan pemilahan, untuk nilai skala keabuan yang dibawah nilai *tresshold* dijadikan 0 atau hitam, sedangkan nilai skala keabuan yang diatas nilai *tresshold* dijadikan 255 atau 1 dalam nilai biner atau putih.

PENERAPAN

Pada dua penelitiannya, Mulyana, (Mulyana, 2013)[14] dan (Mulyana, 2014) [15], binarisasi pada segmentasi obyek citra pada visi komputer merupakan pemisahan obyek pada suatu citra dengan latar belakang nya. Pada umumnya obyek citra diberi nilai 0 atau dijadikan warna hitam, sedangan latar belakang diberi nilai 1 untuk suatu nilai biner atau 255 dalam skala keabuan 8 bit, sehingga menghasilkan warna putih. Pada gambar 2, gambar sebelah kiri merupakan contoh citra dengan hasil binarisasi nya pada citra di sebelah kanannya.

Gambar 2. Contoh Citra Dengan Hasil Binarisasi

Pemilihan Input

Baik pada perceptron dan hebb-rule, input yang digunakan sama. *Input* yang akan dipergunakan untuk menghasilkan citra biner adalah *input* dari intensitas R, G, B, RG, RB dan RGB. Alasan pemilihan *input* ini berdasarkan model RGB yang umum dijumpai pada pengolahan citra. Untuk setiap piksel akan memiliki ketiga *channel* R, G dan B.

Citra biner yang dihasilkan memiliki keterpisahan secara linear, dimana hanya ada dua nilai *output* yaitu hitam (0) dan putih (1) atau direpresentasikan dengan nilai 255 pada skala keabuan. Umumnya obyek citra akan disegmentasi akan memberikan nilai 0 untuk obyek citra, dan nilai 1 untuk latar belakang. *Output* ini mempunyai keterpisahan secara linier. Dengan keterpisahan secara linear tersebut, maka seharusnya algoritma *training* Hebb-Rule yang merupakan algoritma *training* jaringan saraf tiruan yang sangat sederhana cukup memadai untuk dipergunakan.

Proses Pelatihan

Pada setiap obyek yang ditangkap kamera akan menghasilkan lebih dari satu citra dengan posisi obyek yang berbeda-beda. Dengan perbedaan posisi tersebut tentu akan mengakibatkan warna citra yang dihasilkan akan berbeda-beda. Karenanya diperlukan suatu metode pembelajaran untuk mendapatkan bobot yang mampu mengakomodasi proses. Proses ini diperlukan agar dengan beberapa contoh citra yang ditangkap, akan mampu menghasilkan bobot yang dapat digunakan untuk memproses semua citra dengan obyek yang sama.

Pada penelitian ini akan digunakan metode jaringan saraf tiruan dengan alasan sampel yang banyak, sedangkan untuk memproses sebuah citra menjadi citra biner dengan teknik yang sudah dijelaskan di atas diperlukan waktu dan usaha yang banyak. Dan karena setiap citra mempunya variasi bobot R, G, B, RG, RB, GB dan RGB yang berbeda, sehingga sukar untuk menentukan susunan bobot yang sesuai untuk mewakili semua citra dengan obyek yan sama tersebut. Hal ini sesuai dengan kriteria jaringan saraf tiruan yang akan menghasilkan nilai-nilai bobot yang sesuai hanya dengan memberi beberapa sampel.

Citra biner yang dihasilkan memiliki keterpisahan secara linear, dimana hanya ada dua nilai yaitu hitam (0) dan putih (1) atau direpresentasikan dengan nilai 255 pada skala keabuan. Umumnya obyek citra yang akan disegmentasi akan memberikan nilai 0 untuk obyek citra, dan nilai 1 untuk latar belakang.

Dengan keterpisahan secara linear tersebut, maka dapat dipergunakan algoritma *training* hebb rule.

Proses penghitungan *output* dimodifikasi dari persamaan (2) yang disesuai dengan *input* dan bobot yang sudah dirancang pada pembahasan di atas. Rumusan untk menghitung nilai *output* dapat dilihat pada persamaan (9). Dengan ketentuan *output* yang sama, yaitu menjadi dua nilai yaitu kurang dari sama dengan 0 (S=0) yang akan menghasilkan *output* x = 0 dan lebih dari 0 (S>0) yang menghasilkan *output* x = 1

$$S = iR \times wR + iG \times wG + iB \times wB + iRG \times wRG + iRB \times wRB + iRGB \times wRGB + bias \quad (9)$$

keterangan:

- wR : bobot untuk *input* R
- wG : bobot untuk *input* G
- wB : bobot untuk *input* B
- wRG : bobot untuk *input* RG
- wRB : bobot untuk *input* RB
- wGB : bobot untuk *input* GB
- wRGB : bobot untuk *input* RGB
- iR : *input* intensitas R
- iG : *input* intensitas G
- iB : *input* intensitas B
- iRG : *input* intensitas RG
- iRB : *input* intensitas RB
- iBG : *input* intensitas BG
- iRGB : *input* intensitas RGB

Untuk setiap *output* yang tidak sesuai dengan contoh maka bobot berikutnya akan diproses sesuai dengan algoritma training yang digunakan untuk mencari bobot dan bias baru yang disebut dengan proses belajar.

ISSN: 1979-1496

```
if (SOutput <> SLearn) then
begin
  wR := wR + C * (SOutput - SLearn) * iR;
  wG := wG + C * (SOutput - SLearn) * iG;
  wB := wB + C * (SOutput - SLearn) * iB;
  wRG := wRG + C * (SOutput - SLearn) * iRG;
  wRB := wRB + C * (SOutput - SLearn) * iRB;
  wGB := wGB + C * (SOutput - SLearn) * iGB;
  wRGB := wRGB + C * (SOutput - SLearn) * iRGB;
  bias := bias + C * (SOutput - SLearn);
end else
begin
  cocok := true;
end;
```

Gambar 3. Menghitung Bobot dan Bias Baru Dengan Perceptron

Proses Belajar Dengan Algoritma Perceptron

Sesuai dengan algoritma perceptron, maka jika *output* sampel dengan *output* hasil *training* tidak sama akan dilakukan proses penentuan bobot baru. Proses ini dapat dijelaskan program pada gambar 3. Yang dibandingkan adalah S*Output* yang mewakili nilai piksel hasil penghitungan *output*, dan SLearn yang mewakili nilai piksel hasil penghitungan belajar.

Untuk memastikan agar proses looping berhenti maka dibuatlah sebuah variable sebagai *flag* atau penanda bernama 'cocok' yang bertipe Boolean. Variable *flag* ini akan diset dengan nilai FALSE, yang menandakan bahwa piksel yang sedang diuji belum cocok dengan output yang diharapkan. Jika SOuput bernilai sama dengan SLearn, maka variable *flag* ini akan bernilai TRUE. Yang akan membuat *loop* berhenti dan dilanjutkan dengan pengambilan piksel atau sampel lain.

Proses Belajar Dengan Algoritma Hebb-Rule

Sama dengan perceptron, pada algoritma Hebb-Rule, maka jika *output* sampel dengan *output* hasil *training* tidak sama akan dilakukan proses penentuan bobot baru. Proses ini dapat dijelaskan program pada gambar 4.

```
cocok := false;
WHILE (NOT cocok) DO
begin
  .
  .
  .
  if (SOutput <> SLearn) then
  begin
    case RadioGroupTraining.ItemIndex of
      0: begin //HebbRule
      if hitungSampel <= 0 then
        Yrule := -1
      else
        Yrule := 1;
      wR := wR + Yrule * iR;
      wG := wG + Yrule * iG;
      wB := wB + Yrule * iB;
      wRG := wRG + Yrule * iRG;
      wRB := wRB + Yrule * iRB;
      wGB := wGB + Yrule * iGB;
      wRGB := wRGB + Yrule * iRGB;
      bias := bias + Yrule;{}
    end;
    .
    .
    .
  end;
 end else
 begin
  cocok := true;
 end;
end;
```

Gambar 4. Menentukan bobot dan bias baru dengan Hebb-Rule

Untuk mengakomodasi polarisasi dirancang pada program seperti pada gambar 5. Jika hasil penghitungan sampel bernilai 0 atau kurang dari 0 maka nilai *output* yang diinginkan yang diwakili oleh variabel Yrule akan diisi -1, sedangkan jika bernilai lebih dari 1 akan berisi nilai 1.

```
if hitungSampel <= 0 then
  Yrule := -1
else
  Yrule := 1;
```

Gambar 5. Proses polarisasi

Proses Binarisasi.

Dengan menerapkan proses penghitungan *output* berdasarkan bobot dari jaringan saraf tiruan seperti yang tercantum pada rumus (3), dilakukan proses binarisasi dengan bobot-bobot sebagai perbandingan kontribusi ketujuh variasi *channel* RGB.

Proses binarisasi tiap citra dilakukan piksel per piksel. Setiap piksel akan di ambil nilai R, G dan B. Kemudian dari *channel-channel* tersebut akan dihitung skala keabuan yang merupakan variasi dari RG, RB, GB dan RGB.

Karena pada variasi *channel* ini dapat memiliki beberapa kemungkinan satu *channel* saja, kombinasi dua *channel* ataupun kombinasi tiga *channel*, secara umum persamaan tersebut dapat disusun dengan persamaan seperti yang dilakukan pada rumus (11). Proses ini adalah sama dengan proses yang dilakukan pada penelitian sebelumnya yang berjudul "Deteksi Gerakan Tangan secara Optis"

ISSN: 1979-1496

yang dilakukan oleh Mulyana, (Mulyana, 2013).

$$f(x,y) = \sum_{i=1}^{n} C_i(x,y) / n \quad \text{..... (11)}$$

keterangan:

- f(x,y) : piksel abu-abu pada sel(x, y)
- n : jumlah *channel* yang dipilih
- Ci(x,y) : Pilihan *channel* ke i pada sel(x,y)

Karena nantinya proses akan dilakukan berdasarkan algoritma *training* hebb-rule pada jaringan saraf tiruan, maka persamaan (2) yang merupakan penghitungan *output* berdasarkan bobot di adopsi untuk menentukan nilai S. Sehingga nilai S akan mempunyai variasi -1, 0 dan 1. Karena pada proses ini hanya diperlukan dua kondisi yaitu hitam dan putih, maka untuk -1 dan 0 akan diperlakukan sama, sehingga keterpisahan linear akan terpolarsasi menjadi dua nilai yaitu kurang dari sama dengan 0 (S=0) yang akan menghasilkan *output* x = 0 dan lebih dari 0 (S>0) yang menghasilkan *output* x = 1. Persamaan (2) tersebut diadopsi dengan ketentuan:

- *input* a1 = intensitas R
- *input* a2 = intensitas G
- *input* a3 = intensitas B
- *input* a4 = intensitas RG
- *input* a5 = intensitas RB
- *input* a6 = intensitas GB
- *input* a7 = intensitas RGB

Proses ini diterapkan pada setiap obyek yang ditangkap kamera akan menghasilkan lebih dari satu citra dengan posisi obyek yang berbeda-beda. Dengan perbedaan posisi tersebut tentu akan mengakibatkan warna citra yang dihasilkan akan berbeda-beda. Karenanya diperlukan suatu metode pembelajaran untuk mendapatkan bobot yang mampu mengakomodasi proses. Proses ini diperlukan agar dengan beberapa contoh citra yang ditangkap, akan mampu menghasilkan bobot yang dapat digunakan untuk memproses semua citra dengan obyek yang sama. Proses ini diterapkan pada program yang dapat dilihat pada gambar 5.

```
hitung := iR * wR + iG * wG + iB * wB + iRG * wRG
          + iRB * wRB + iGB * wGB + iRGB * wRGB;
if round(hitung) > 0 then
begin
  ImageSelected.Canvas.Pixels[x,y] := RGB(255,255,255);
end else
begin
  ImageSelected.Canvas.Pixels[x,y] := RGB(0,0,0);
end;
```

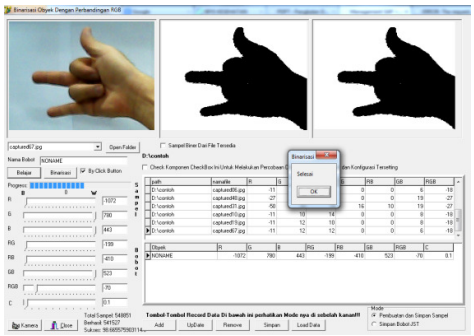
Gambar 5. Menentukan *Output*

HASIL DAN PEMBAHASAN

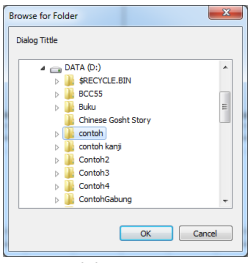
Untuk menunjang penelitian dibuat perangkat lunak pemroses binarisasi dengan mempergunakan *input* ketujuh variasi intensitas R, G, B, RG, RB GB dan RGB dengan kedua algoritma training hebb-rule dan perceptron sebagai pilihan.

Perangkat lunak dibuat dengan interface seperti pada gambar 6. Untuk menampilkan frame-frame citra yang diproses, disediakan tiga panel tampilan citra:

- Bagian tampilan citra yang akan dibinarisasi
- Bagian tampilan citra hasil binarisasi
- Bagian tampilan citra untuk menampilkan hasil belajar.



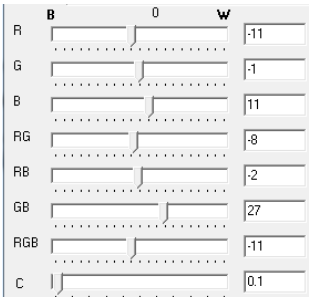
Gambar 6. Tampilan Jendela Program



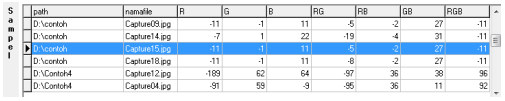
Gambar 7. Pemilihan Folder Kumpulan Citra

Tabel 1. Data Percobaan

Folder	Jumlah Sampel Citra	Kondisi	Mulai	Selesai	waktu	metode	Sampel piksel	Iterasi
Contoh	5	Homogen	10:29:42	10:31:31	0:01:49	Hebb-Rule	386,805	387,900
Contoh	5	Homogen	10:32:32	10:34:22	0:01:50	Perceptron	386,805	387,915
Contoh 2	5	Homogen	2:45:51	2:47:51	0:02:00	Hebb-Rule	387,900	388,758
Contoh 2	5	Homogen	2:47:59	2:49:49	0:01:50	Perceptron	387,900	388,758
Contoh 3	10	Homogen	10:36:14	10:40:13	0:03:59	Hebb-Rule	773,610	775,541
Contoh 3	10	Homogen	10:41:28	10:45:11	0:03:43	Perceptron	773,610	775,541
Contoh 4	6	Homogen	10:49:27	10:51:39	0:02:12	Hebb-Rule	464,166	466,158
Contoh 4	6	Homogen	10:47:14	10:49:18	0:02:04	Perceptron	464,166	466,217
Gabungan	26	Heterogen	11:05:53	11:15:34	0:09:41	Hebb-Rule	2,011,386	2,016,590
Gabungan	26	Heterogen	10:54:37	11:04:25	0:09:48	Perceptron	2,011,386	2,016,678



Gambar 8. Pengaturan Bobot pada masing-masing *input*



Gambar 9. Data Grid Pencatat Konfigurasi Bobot

Setiap citra dengan obyek yang sama akan dikumpulkan dalam satu folder, untuk mengambil citra-citra tersebut cukup dengan memilih folder. Proses pemilihan folder dapat dilihat pada gambar 7.

Pengaturan nilai bobot dapat dilihat pada gambar 8. Pengaturan bobot ini nanti akan berpengaruh pada citra biner yang dihasilkan. Berbeda dengan perceptron, pada hebb-rule tidak diperlukan nilai kecepatan belajar karena itu pengaturan kecepatan belajar tidak perlu diatur.

Konfigurasi bobot berikut path folder dan nama filenya akan disimpan. Data yang tersimpan akan ditampilkan pada data grid sampel yang nampak pada gambar 9.

Hasil Pengujian

Pengujian teknik ini dilakukan dengan mempergunakan perangkat lunak yang sudah dibuat. Dimana pengujian dilakukan dengan kedua algoritma training. Yang diuji pada keduanya adalah jumlah iterasi dan waktu.

Hasil pengujian dapat dilihat pada Tabel 1, yang merupakan tabel data percobaan dengan algoritma *training* perceptron dan algoritma *training* hebb-rule.

Percobaan dilakukan terhadap 42 citra dengan posisi yang berbeda-beda. Citra-citra tersebut meskipun menangkap obyek yang sama, tetapi karena adanya posisi cahaya dan jarak dengan kamera berpengaruh terhadap warna dari obyek citra tersebut. Ada bagian yang semakin terang ada bagian yang semakin gelap. Karena nya setelah didapat beberapa model binarisasi dari beberapa citra pada masing-masing ketegori, citra hasil binarisasi berikut komposisi bobot-bobot nya disimpan. Nantinya dilakukan *training* sehingga menghasilkan variasi bobot-bobot yang dapat mewakili semua citra tersebut.

SIMPULAN

Berdasarkan penelitian Penerapan Hebb-Rule Pada Segmentasi Obyek Citra

Berdasarkan Perbandingan *Channel* RGB, dapat disimpulkan:

1. Iterasi Algoritma Hebb-Rule relative lebih sedikit daripada perceptron.
2. Nilai bobot yang dihasilkan algoritma Hebb-Rule lebih besar daripada Perceptron.
3. Dengan iterasi Algoritma Hebb-rule yang relatif lebih sedikit daripada perceptron, maka sama seperti algoritma *training* perceptron, algoritma *training* hebb-rule juga layak dipergunakan dalam proses binarisasi dengan jaringan saraf tiruan.

DAFTAR PUSTAKA

[1] Ahuja, Vicky., Anand, Vivek., Gandhi, Varun., Yadav, Varsha., "*Artificial neural network: all about artificial neural world, Indian Journal of Engineering*", Volume 6, Number 15, November 2013 (ISSN 2319 – 7757 EISSN 2319 –7765), page 6-8

[2] Aviv, Rotem., "Algorithms for testing connectivity - Implementation on binary images", Tel-Aviv University, Tel-Aviv, 2011. p:14

[3] Chan, Tony F., Esedo¯glu, Selim., Nikolova, Mila., "*Finding The Global Minimum For Binary Image Restoration*", IEEE, 2005 (pp: I21-I24)

[4] Chen, Tao., Takagi, Mikio., "*Image Binarization By Back Propagation Algorithm*", International Society for Photogrammetry and Remote Sensing XXIX, 1992, August 2-14, 1992, Washington, D.C., USA. p:345-349

[5] Desiani, Anita., Arhami, Muhammad., Konsep Kecedasan buatan, Andi Offset, Yogyakarta, 2006.

[6] Gonzalez and Woods, Digital image processing, 2nd Edition, prentice hall, 2002

[7] Hermawan, Arif., Jaringan Saraf Tiruan, Andi Offset, Yogyakarta, 2006.

[8] Huyck, Christian R., Mitchell, Ian G., "*Compensatory Hebbian learning for categorisation in simulated biological neural nets*", Journal: Biologically Inspi*Red* Cognitive Architectures, Vol 6, 2013, (ISSN: 2212-683X) page 3-7

[9] Ibraheem, Noor A., Hasan, Mokhtar M., Khan, Rafiqul Z., Mishra, Pramod K., "*Understanding Color Models: A Review*", ARPN Journal of Science and Technology VOL. 2, NO. 3, April 2012, (p265-275)

[10] Khardon, Roni., Wachman, Gabriel., "*Noise Tolerant Variants of the Perceptron Algorithm*", Journal of Machine Learning Research 8 (2007) 227-248 Submitted 11/05; Revised 10/06; Published 2/07

[11] Low, Adrian., "*Introductory Computer Vision and Image Processing*", McGraw-Hill, Berkshire, UK, 1991

[12] Mulyana, Teady M.S., "*Penggunaan Nilai Skala Keabuan dari Citra Watermark sebagai Cetak Biru Dari Visible Watermarking*", Procedding Seminar Nasional Informatika 2013, 18 Mei 2013, UPN Yogyakarta

[13] Mulyana, Teady.M.S., Suherman, Anggie., 2013, "*Segmentasi Objek Citra Berdasarkan Perbandingan Channel RGB"*, Bunda Mulia University, Jakarta. (Laporan Penelitian - not published)

[14] Mulyana, Teady.M.S., Heriyono, Valerian., "Penerapan Hebb-Rule Pada Segmentasi Obyek Citra Berdasarkan Perbandingan *Channel* RGB", Bunda Mulia University, Jakarta, 2014. (Laporan Penelitian - not published)

[15] Nobuyuki. Otsu, "*A threshold selection method from gray level histograms*," IEEE Trans. Systems, Man and Cybernetics, vol. 9, pp. 62–66, 1979.

[16] Mukherjee, Aroop. Kanrar, Soumen., "*Enhancement of Image Resolution by Binarization"*, International Journal of Computer Applications (ISSN: 0975–8887), Volume 10 – No.10, November 2010 15, p:15-19

[17] Pugh, Justin K., Soltoggio, Andrea,, and Stanley, Kenneth O., "*Real-time Hebbian Learning from Autoencoder Features for Control Tasks*", Proceeding of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14).Cambridge, MA: MIT Press, 2014.

[18] Ridler, T., Calvard, S., "*Picture thresholding using an iterative selection method*," IEEE Transactions on Systems, Man and Cybernetics, vol. 8, no. 8, pp. 630–632, Aug. 1978

[19] Sathasivam, Saratha., "*Learning Rules Comparison in Neuro-Symbolic*

Integration", International Journal of Applied Physics and Mathematics, Vol. 1, No. 2, September 2011, AbuDhabi University, UAE (ISSN: 2010-362X) page 129-132

[20] Sauvola, J., Kinen, M. Pietika., "*Adaptive document image binarization*", Pattern Recognition Vol 33, Pattern Recognition Society. Elsevier Science Ltd, 1999, pp:225-236

[21] Sutoyo, T., Mulyanto, E., Suhartono V., Nurhayanti OD., Wijanarto., "Teori Pengolahan Digital", Andi Offset, Yogyakarta, 2009.

[22] Wen, Che-Yen., Chou, Chun-Ming, "Color Image Models and its Applications to Document Examination", Forensic Science Journal. pp. 23-32 .Vol. 3(1). 2004.