OPTICAL CHARACTER RECOGNITION MENGGUNAKAN *SELF ORGANIZING MAP*

Ali
Teknik Informatika, Institut Teknologi Nasional Malang
aliabunn@gmail.com

ABSTRAK

Pengenalan karakter teks secara digital merupakan bagian dari ilmu computer *vision*. Dalam pengenalan karakter dibutuhkan metode kecerdasan buatan supaya dapat membedakan setiap karakternya. *Optical Character Recognition* (OCR) merupakan salah satu metode pengenalan karakter cetak dengan menggunakan kecerdasan buatan sehingga teks dapat dikenali dengan beberapa tahapan. Penulis memilih metode *Self Organizing Map* (SOM) karena menurut penulis metode tersebut dapat diterapkan untuk mengenali karakter dan SOM merupakan salah satu kecerdasan buatan yang tidak membutuhkan pengawasan.

Algoritma *Self Organizing Map* (SOM) atau yang sering disebut dengan Jaringan Syaraf Tiruan Kohonen merupakan suatu metode jaringan syaraf tiruan yang diperkenalkan oleh Professor Teuvo Kohonen pada tahun 1981. Jaringan Kohonen merupakan salah satu bentuk topologi dari *Unsupervised Artificial Neural Network* (*Unsupervised ANN*) di mana dalam proses pelatihannya tidak memerlukan pengawasan (target output). Jaringan Kohonen/SOM digunakan untuk mengelompokkan (*clustering*) data berdasarkan karakteristik/fitur-fitur data. Penelitian ini menerapkan SOM dalam aplikasi berbasis Android dengan sistem operasi minimum v4.4.2 (KitKat) dan maksimum v5.1 (Lollipop).

Penelitian ini lebih berfokus kepada pengenalan karakter teks. Masukan dalam penelitian ini berupa citra yang diambil dari *smartphone*. Segmentasi yang digunakan untuk memisahkan obyek dengan latar belakang citra yaitu *Connected Component Labeling* (CCL). Masukan dalam pengenalan pola SOM berupa 2 buah vektor yang berisi jumlah citra berwarna hitam. Iterasi dilakukan sebanyak 100 kali sehingga hasil akhir *clustering* merupakan hasil pengenalan karakter. Data *training* menggunakan jenis *font* Arial berukuran 14 dengan *style* tebal (*bold*) sebanyak masing-masing huruf 3 data. Persentase keberhasilan pengenalan karakter yaitu 42,304 % dengan kecepatan rata-rata pengenalan karakter 3.74 detik/karakter.

Kata kunci : *Optical Character Recognition, Self Organizing Map, Jaringan Syaraf Tiruan Kohonen, Android, Connected Component Labelling, Clustering*

1. PENDAHULUAN

1.1. Latar Belakang

Optical Character Recognition (OCR) sudah mulai digunakan dalam pengembangan aplikasi. Namun dalam pengujian hasilnya masih memiliki beberapa kendala sehingga hasil pengenalan teks tidak dapat dikenali secara keseluruhan. Biasanya kesalahan terletak pada data pelatihan atau masukan yang kurang cocok sehingga hasil dari pelatihan tidak sama dengan huruf aslinya. Terdapat beberapa *library* tambahan yang berfungsi untuk mengenali teks cetak seperti Tesseract OCR yang dikembangkan oleh Google tetapi penelitian tentang pengenalan karakter tetap dilakukan untuk mengetahui sistem pengenalan karakter karena *library* merupakan alat bantu yang tidak diketahui proses pengolahannya. Penerapan OCR membutuhkan beberapa proses yang beragam sesuai dengan masukan pada metode yang digunakan seperti segmentasi baris, segmentasi kolom sampai segmentasi ukuran teks. Selain proses yang beragam, metode pengenalan teks dalam citra juga harus diperhatikan.

Clustering merupakan salah satu metode pengelompokan data berdasarkan kedekatan karakteristik. *Self Organizing Map* (SOM) merupakan salah satu metode pengelompokan (*clustering*) yang sering digunakan untuk melakukan ekstraksi fitur (Lestari, 2010). Pramesti (2014) melakukan sebuah penelitian yang juga merupakan OCR dengan menggunakan metode *neural network backpropagation*. Penelitian tersebut merupakan penelitian yang menggunakan 2 jenis metode kecedasan buatan (*hybrid*) yaitu logika *fuzzy* (sebagai proses klasifikasi segmen) dan *neural network backpropagation* (sebagai proses pengenalan karakter). Hasil persentase keberhasilan pengujian penelitian tersebut yaitu 80.12 %.

Penelitian ini membutuhkan proses ekstraksi fitur dari citra supaya citra dapat dikelompokkan berdasarkan kemiripan karakteristik dengan data *training*. Menurut penulis, SOM merupakan salah satu metode yang tepat untuk digunakan sebagai kecerdasan buatan pengenalan karakter karena SOM merupakan salah satu kecerdasan buatan yang tidak memerlukan pengawasan / *target output (unsupervised artificial neural network).* Selain itu SOM juga mudah dipahami dan diimplementasikan ke dalam sebuah kode / bahasa pemrograman komputer, dengan demikian penulis membuat penelitian tentang penerapan SOM dalam sebuah OCR. Penulis mengembangkan

aplikasi OCR berbasis Android karena OCR membutuhkan akuisisi citra dari kamera digital, dengan demikian penulis menggunakan Eclipse IDE sebagai perangkat lunak pengembang aplikasi OCR.

1.2. Rumusan Masalah

Berdasarkan pada latar belakang masalah di atas maka didapatkan rumusan masalah yang akan dibahas yaitu bagaimana cara mengenali karakter teks menggunakan metode SOM?

1.3. Tujuan

Adapun tujuan dibuatnya penelitian ini adalah dapat mengenali karakter teks menggunakan metode SOM.

1.4. Batasan Masalah

Pada penelitian ini terdapat beberapa batasan masalah yang dihadapi, yaitu:

1. Penelitian menggunakan bahasa pemrograman Java.
2. Penelitian diterapkan pada *smartphone* berbasis Android dengan sistem operasi minimum v4.4.2 (KitKat) dan maksimum v5.1 (Lollipop).
3. Penelitian menggunakan Eclipse IDE untuk mengembangkan aplikasi Android.
4. Menggunakan *Optical Character Recognition* sebagai metode pengambilan teks pada citra.
5. Menggunakan *Self Organizing Map* sebagai metode pengenalan teks.
6. Huruf harus kapital.
7. *Font* menggunakan Arial dengan ukuran 14 dan *style* tebal (*bold*).
8. Latar belakang gambar harus berwarna putih.

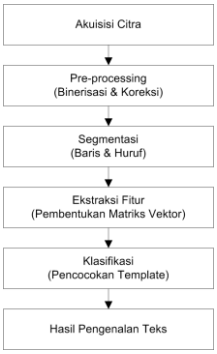
2. TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Pramesti (2014) melakukan sebuah penelitian tentang penggunaan OCR menggunakan metode *neural network backpropagation*. Penelitian tersebut menggunakan citra RGB sebagai masukan. Untuk proses ekstraksi fitur penelitian terkait menggunakan pembacaan bentuk geometri setiap segmen dalam sebuah obyek yang sudah dilakukan segmentasi sebelumnya. Dalam proses klasifikasi sebuah segmen penelitian tersebut menggunakan metode fuzzy sehingga setiap bagian dari segmen dapat dikenali seperti bentuk garis vertical, horizontal dan diagonal. Proses ini akan mendeteksi obyek yang saling terhubung (*loop*) dan tidak mengamatinya. Keluaran dari proses pengenalan bentuk geometri berupa persamaan garis pokok dan keluaran berupa matrik berukuran 1x4 di mana kolom pertama menunjukkan angka yang mewakili setiap bentuk geometri, kolom kedua adalah ukuran segmen, kolom ketiga adalah posisi secara vertical dan kolom keempat merupakan posisi secara horizontal. Matrik tersebut akan digunakan sebagai masukan dari metode klasifikasi *neural network backpropagation* dengan 400 citra yang akan dilakukan pelatihan. Keluaran dari klasifikasi ini berupa nilai biner yang berisi karakter ASCII. Hasil persentasi keberhasilan pengujian penelitian tersebut yaitu 80.12 %.

2.2. Optical Character Recognition

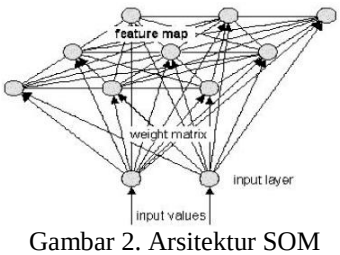
Optical character recognition (OCR) merupakan salah satu dari banyak penerapan teknologi terhadap pengenalan yang menggunakan kecerdasan buatan. Namun kecanggihan perangkat keras maupun perangkat lunak yang menerapkan OCR belum mampu menggantikan kemampuan membaca manusia (Chandarana, 2014). OCR sendiri berusaha untuk mengenali teks pada sebuah kalimat. Teks tersebut merupakan teks hasil cetakan, bukan tulisan tangan. Langkah penerapan OCR dalam sebuah aplikasi dapat dilihat pada Gambar 1.



Gambar 1. Langkah penerapan OCR

2.3. Self Organizing Map

Jaringan syaraf tiruan kohonen atau *self organizing map* (SOM) merupakan salah satu cabang dari *unsipervised artificial neural network (unsupervised* ANN*). Unsupervised* ANN merupakan metode kecerdasan buatan berbasis jaringan syaraf tiruan yang tidak membutuhkan pengawasan terhadap proses pelatihannya (*target output*). SOM digunakan untuk mengelompokkan (*clustering*) data berdasarkan dengan karakteristik / fitur data (Lestari, 2010). Arsitektur SOM dapat dilihat pada Gambar 2.



Gambar 2. Arsitektur SOM

SOM terdiri dari suatu lapisan yang berisi neuron-neuron (*input layer*) di mana neuron tersebut akan menyusun dirinya hingga menjadi beberapa kelompok (*cluster*) sesuai dengan nilai masukannya. Neuron tersebut berupa nilai dari kriteria tertentu (*input values*) yang sebelumnya sudah ditentukan dan

dilakukan ekstraksi fitur. Proses penyusunan neuron dihitung berdasarkan vektor bobot (*weight matrix*) yang setiap iterasinya akan menyesuaikan sehingga vektor bobot yang memiliki jarak paling dekat (sesuai dengan pola masukan) akan menjadi pemenang. Neuron keluaran berupa kumpulan dari neuron pemenang (*winning neuron* pada proses penyusunan neuron) yang dikelompokkan berdasarkan kelas (*cluster*) yang memiliki jarak paling dekat (kemiripan karakteristik) terhadap neuron masukan lainnya.

Algoritma SOM yaitu :

1. Inisialisasi neuron masukan $x_1, x_2, \ldots x_i$.
2. Inisialisasi neuron keluaran $y_1, y_2, \ldots y_i$.
3. Menentukan bobot (w_{ij}) antara neuron masukan dengan neuron keluaran dengan nilai antara 0 samapi 1 (*random*).
4. Inisialisasi bobot bias (b_i) menggunakan rumus Persamaan 2.1.

$$b_i = e^{\left(1-\ln\left(\frac{1}{K}\right)\right)} \quad (2.1)$$

K : Jumlah kelas (neuron keluaran).

5. Mengulangi langkah 6 sampai 12 hingga bobot tidak berubah atau iterasi (*epochs*) sudah mencapai batas sehingga keluaran telah konvergen.
6. Memilih salah satu masukan dari vektor masukan yang ada.
7. Menghitung jarak antar masukan terhadap bobot (D_i) dengan setiap neuron masukan menggunakan Persamaan 2.2.

$$D_i = \sum_{i=1}^{n} (w_{ij} - x_i)^2 \quad (2.2)$$

8. Menghitung nilai jarak (a_i) dengan menggunakan Persamaan 2.3.

$$a_i = -D_i + b_i \quad (2.3)$$

9. Mencari nilai a_i terbesar.
10. Mengatur keluaran neuron (y_i) dengan ketentuan pemenang = 1 dan bukan pemenang = 0.
11. Memperbarui bobot (w_{ij}) dengan perhitunagn menggunakan Persamaan 2.4.

$$w_{ij} = w_{ij} + \alpha\left(x_i - w_{ij}\right) \quad (2.4)$$

12. Memperbarui bobot bias (b_i) dengan perhitunagn menggunakan Persamaan 2.5 dan Persamaan 2.6.

$$c(i) = (1-\alpha)e^{(1-\ln(b(i)))} + \alpha a(i) \quad (2.5)$$

$$b(i) = e^{(1-\ln(c(i)))} \quad (2.6)$$

13. Menyimpan bobot yang telah konvergen (Anis, Isnanto, 2014).

2.4. Pengolahan Citra Digital

Pengolahan citra digital merupakan sebuah disiplin ilmu yang mempelajari tentang teknik mengolah citra. Citra tersebut adalah gambar diam maupun gambar bergerak yang diambil melalui kamera. Sedangkan digital adalah cara pengolahan yang menggunakan *computer* (Sutojo *et all.* 2009).

2.5. Smoothing

Penghalusan (*smoothing*) merupakan proses di mana citra yang masih memiliki gangguan (*noise*) dihaluskan sehingga gangguan tidak terdapat lagi dalam citra (Rindiani, Nisa, 2015). Gangguan merupakan bagian yang dapat mempengaruhi/mengubah hasil dari sebuah proses sehingga hasil tidak maksimal/tercapai. Sebagai contoh gangguan dalam citra adalah titik yang sebenarnya tidak terdapat pada obyek pengambilan citra (obyek sebenarnya) namun terdapat pada citra digital. Biasanya titik tersebut muncul ketika citra diolah seperti penajaman citra digital.

2.6. Grayscale

Citra keabuan (*grayscale image*) merupakan citra yang hanya memiliki satu jenis intensitas warna, yaitu keabuan. Intensitas keabuan memiliki 256 jenis warna di antara warna hitam dan putih. Nilai intensitas keabuan di mulai dari 0 – 255 di mana nilai 0 merupakan warna hitam dan 255 merupakan warna putih (Rindiani, Nisa, 2015).

2.7. Threshold

Pengambangan citra (*image thresholding*) merupakan sebuah proses memisahkan latar belakang dengan obyek yang akan diamati. Proses ini menghasilkan citra dengan hanya dua jenis informasi, yaitu pixel latar dan pixel obyek. Dengan demikian citra keluaran akan menjadi citra biner (Huang, Chau, 2008). Pengambangan citra dilakukan berdasarkan dengan karakteristik nilai intensitas pixel.

2.8. Adaptive Thresholding

Sebuah citra integral (tabel penjumlahan area) adalah sebuah cara yang dapat digunakan untuk mengubah nilai pixel menjadi angka yang dapat dihitung melalui sebuah matrik (kernel) dalam sebuah gambar. Sebagai contoh penggunaan citra integral sebagai pengenalan wajah dan pemetaan pola (Bradley, Roth, 2007). Perhitungan citra integral membutuhkan penjumlah nilai pixel dengan pixel tetangga ($f(x,y)$) dari sebuah pixel yang sedang diamati ($I(x,y)$). Semua pixel akan diamati sehingga dapat dihitung secara konstan menggunakan Persamaan 2.7.

$$I(x,y) = f(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1) \quad (2.7)$$

Sedangkan untuk menghitung nilai dari pixel tetangga ($f(x,y)$) dari posisi kiri atas (x_1,y_1) sampai posisi kanan bawah (x_2,y_2) dapat dihitung secara konstan menggunakan Persamaan 2.8.

$$I(x,y) = f(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1) \quad (2.8)$$

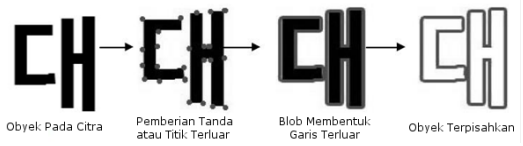
2.9. Segmentasi

Segmentasi citra merupakan sebuah proses yang digunakan untuk menghasilkan sekumpulan wilayah yang dikelompokkan dalam citra / kontur yang diekstrak dari citra. Terdapat dua pendekatan utama dalam segmentasi

citra yaitu berdasarkan deteksi tepi (*edge detection*) dan berdasarkan wilayah (*region-based*). Pada segmentasi berbasis deteksi tepi, citra dibagi berdasarkan diskontinuitas di antara sub-wilayah (*sub-region*). Sedangkan pada segmentasi berbasis wilayah, citra dibagi berdasarkan keseragaman sub-wilayah (Murinto, Agus, 2011).

2.10. Connected Component Labelling

Connected Component Labeling (CCL) merupakan sebuah proses segmentasi citra di mana pixel yang sudah ditandai sebagai obyek akan dipisahkan dari latar berdasarkan titik terluar (*outline*). Proses pemisahan komponen pada obyek akan terus dilakukan hingga titik akhir merupakan titik awal pemisahan. Titik yang sudah ditandai tersebut disebut dengan *blob* (Rindiani, Nisa, 2015). Blob menghasilkan garis terluar (*outline*) dari obyek sehingga obyek dikenali dan dapat dilakukan pengamatan sesuai dengan kebutuhan. Proses segmentasi menggunakan metode CCL dapat dilihat pada Gambar 3.



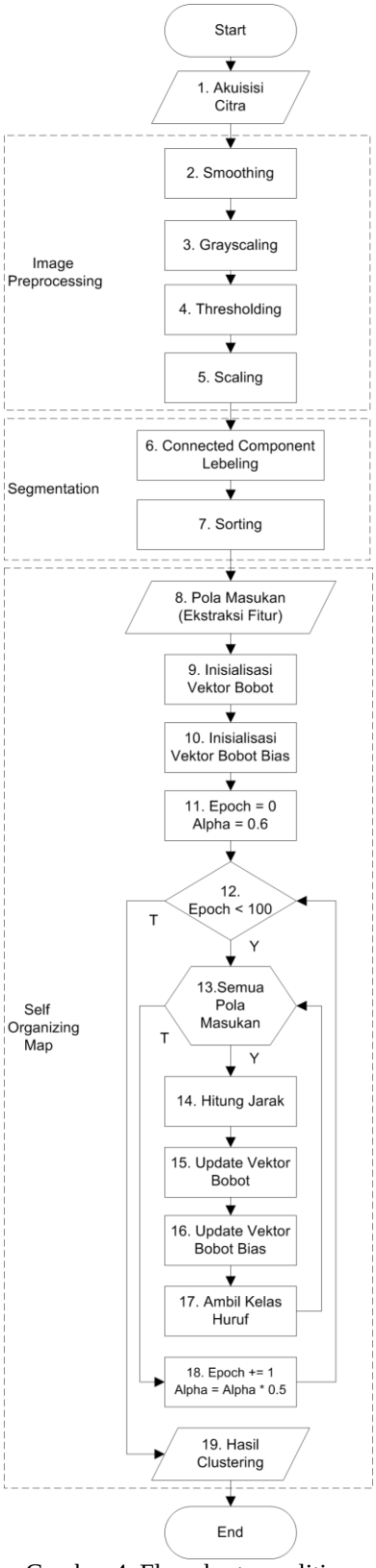
Gambar 3. Proses segmentasi CCL

2.11. Eclipse IDE

Eclipse IDE merupakan sebuah perangkat lunak yang dapat digunakan untuk mengembangkan aplikasi *platform independent* (dapat dijalankan di semua platform seperti Microsoft Windows, Linux dan Mac OS X). Selain itu Eclipse juga merupakan *free software* (perangkat lunak yang dapat digunakan secara gratis oleh semua orang) dengan kemampuan dapat dikembangkan dengan *plug-in*. Eclipse IDE sendiri mendukung beberapa bahasa pemrograman seperti Java, C/C++, Python, PHP, dan lain sebagainya (Sundari, 2012).

3. METODE PENELITIAN

Flowchart penelitian merupakan gambaran daru alur sistem yang dikerjakan secara keseluruhan dalam suatu proses tertentu dan menjelaskan prosedur-prosedur yang ada dalam system. Flowchart tersebut dapat dilihat pada Gambar 4.



Gambar 4. Flowchart penelitian

Dalam penelitian ini penulis membagi penelitian secara garis besar menjadi 3 bagian, yaitu bagian *image preprocessing*, *segmentation* dan SOM. Pada bagian *image preprocessing* terdapat 4 proses yang mengolah citra sehingga citra dapat dibedakan antara obyek dengan latar belakang. Pada bagian *segmentation* terdapat 2 proses yang mengolah citra sehingga obyek dalam citra menjadi sebuah data yang digunakan sebagai masukan dalam proses selanjutnya (SOM). Pada bagian SOM terdapat 12 proses yang mengolah data dari citra sehingga menjadi sebuah kata dalam bentuk *string*. Penjelasan tentang flowchart penelitian yaitu :

1. Akuisisi Citra

Dalam proses ini masukan berupa gambar yang diambil melalui kamera *smartphone*. Gambar tersebut berupa citra RGB dengan resolusi yang besar (tergantung dari ukuran resolusi perangkat kamera). Gambar yang sudah diambil (*capture*) tersebut selanjutnya dipotong (*crop*) dan hanya diambil bagian yang memiliki tulisan yang akan dikenali. Pemotongan gambar tersebut dilakukan secara manual oleh pengguna. Keluaran dari proses ini adalah citra yang sudah terpotong sehingga hanya tersisa bagian yang memiliki tulisan / teks.

2. *Smoothing*

Pada proses ini masukan berupa citra yang sudah terpotong. Selanjutnya citra akan dihaluskan dengan cara melakukan konvolusi (perkalian pixel) pada citra dengan menggunakan matrik ordo 3x3 dengan nilai seperti pada Gambar 5.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 5 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Gambar 5. Nilai matrik penghalusan

Keluaran dari proses ini yaitu citra yang sudah dihaluskan.

3. *Grayscaling*

Pada tahap ini masukan berupa citra yang sudah dihaluskan. Selanjutnya citra yang masih memiliki 3 intensitas warna (RGB) diubah menjadi hanya 1 intensitas warna (*grayscale*) dengan cara menjumlah nilai intensitas warna merah (*red*), hijau (*green*), biru (*blue*) dan membaginya dengan 3 seperti pada Persamaan 3.1.

$$Grayscale = \frac{R + G + B}{3} \qquad (3.1)$$

Keluaran dari proses ini yaitu citra yang sudah memiliki satu intensitas warna yaitu keabuan (*grayscale*).

4. *Thresholding*

Pada tahap ini masukan berupa citra yang sudah memiliki satu intensitas warna yaitu keabuan. Selanjutnya citra diubah menjadi biner dengan ketentuan citra yang memiliki nilai pixel = 0 merupakan latar belakang dan citra yang memiliki nilai pixel = 1 merupakan obyek. Proses ini menggunakan metode *adaptive thresholding*. Keluaran pada tahap ini yaitu berupa citra yang sudah berisi hanya 2 warna, hitam dan putih (biner).

5. *Scaling*

Pada proses ini masukan berupa citra yang sudah biner. Selanjutnya citra yang masih memiliki ukuran besar dilakukan pengecilan resolusi (*scaling*) dengan cara menghitung resolusi dengan rumus seperti pada Persamaan 3.2.

$$Width = Height\left(\frac{Width}{Height}\right) \qquad (3.2)$$

Keluaran dari proses ini yaitu citra yang sudah memiliki panjang resolusi 150 pixel.

6. *Connected Component Labeling*

Pada proses ini masukan berupa citra yang memiliki panjang resolusi 150 pixel. Selanjutnya citra dilakukan segmentasi dengan metode CCL sehingga obyek (nilai pixel 1) dikenali dan dipisahkan setiap obyeknya. Keluaran dari proses ini berupa beberapa citra yang terdiri pada setiap citranya yaitu satu obyek.

7. *Sorting*

Pada tahap ini masukan berupa beberapa citra berisi obyek yang masih belum tersusun. Selanjutnya citra disusun sesuai dengan urutannya dengan cara menyusun sesuai nilai awal pada sumbu x citra menggunakan metode penyusunan gelembung (*bubble sort*). Proses penyusunan tersebut dapat dilihat pada Gambar 6.

```
5  3  4  2  1
1  5  3  4  2
1  2  5  3  4
1  2  3  5  4
1  2  3  4  5
```

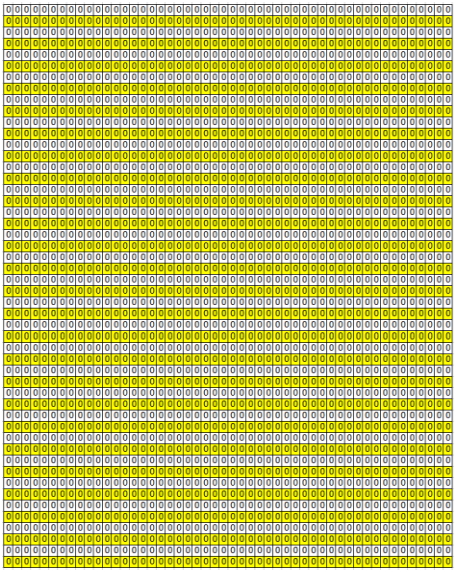
Gambar 6. Penyusunan citra

Keluaran dari tahap ini yaitu beberapa citra obyek yang sudah tersusun.

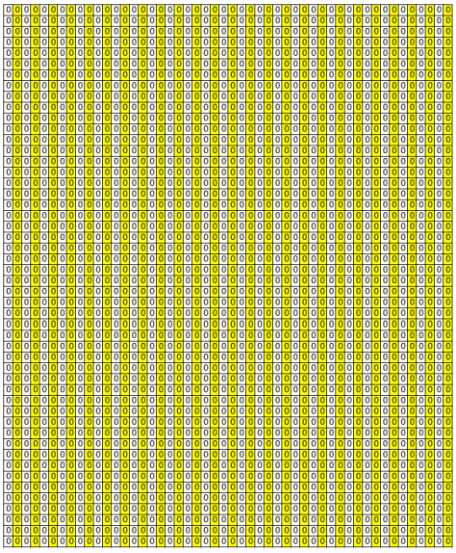
8. *Pola Masukan (Ekstraksi Fitur)*

Pada tahap ini masukan (ekstraksi fitur) berupa data pelatihan dan data masukan. Data tersebut berupa *array* vektor sebanyak 50 data pada setiap polanya. Data pelatihan (*training*) terdiri dari 78 pola yaitu pola huruf a sampai z (setiap huruf memiliki 3 jenis pola yang berbeda) dan data masukan terdiri dari 1 pola yaitu pola dari citra yang sedang diamati. *Array* vektor sebanyak 50 diperoleh dari penjumlahan *pixel* dalam sebuah sumbu (x dan y) pada citra lalu dibagi 5, proses ini dilakukan pada setiap sumbu x dan sumbu y (25 sumbu x dan 25 sumbu y). 25 sumbu yang dipilih pada sumbu x adalah sumbu dengan indeks ke 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, sedangkan 25 sumbu

yag dipilih pada sumbu y adalag sumbu dengan indeks ke 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49. Indeks dari sumbu yang dipilih dimulai dari 0 samapi 49 (50 baris *pixel*). Setiap sumbu memiliki panjang 50 *pixel* (data), baik pada sumbu x maupun sumbu y. Nilai biner dari *pixel* dijumlah dari sumbu indeks ke 1 *pixel* ke 1 (pertama) sampai sumbu indeks ke 1 *pixel* ke 50 (terakhir), lalu dibagi 5. Hasil dari pembagian tersebut merupakan 1 data dari *array* vektor pola masukan. Penjumlahan nilai biner dari *pixel* tersebut dilakukan sebanyak 25 sumbu pada koordinat x dan 25 sumbu pada koordinat y sehingga terdapat hasil bagi dari setiap penjumlahan *pixel* pada sumbu x dan y sebanyak 50 data. Penjelasan inisialisasi vektor dapat dilihat pada Gambar 7 dan Gambar 8.



Gambar 7. Pixel sumbu x



Gambar 8. Pixel sumbu y

Sumbu dengan warna kuning (pada Gambar 7 dan Gambar 8) merupakan sumbu yang dilakukan penjumlahan nilai *pixel* dalam sumbu tersebut lalu dibagi dengan angka 5 pada setiap sumbunya. Gambar 7 merupakan pengambilan data pada sumbu x yang terpilih (25 data) sedangkan Gambar 8 merupakan pengambilan data pada sumbu y yang terpilih (25 data).

9. Inisialisasi Vektor Bobot

Pada tahap ini dilakukan deklarasi sekaligus inisialisasi variabel yang berisi nilai random di antara 0 samapi 1. Variabel ini merupakan variabel penting karena pada setiap iterasinya variabel akan diperbarui sesuai dengan jarak terdekatnya.

10. Inisialisasi Vektor Bobot Bias

Pada tahap ini dilakukan deklarasi sekaligus inisialisasi variabel yang berisi nilai pecahan sesuai dengan jumlah kelas (neuron keluaran). Nilai tersebut yaitu 70.67532753993514. variabel ini juga merupakan variabel penting yang nantinya akan berubah sesuai dengan neuron pemenang pada setiap iterasinya.

11. Inisialisasi Epoch dan Alpha (Learning Rate)

Pada tahap ini dilakukan deklarasi sekaligus inisialisasi variabel *epoch* dengan nilai = 0. Variabel ini merupakan variabel penentu berakhirnya iterasi. Variabel ini akan terus bertambah ketika proses latihan selesai dilakukan. Selain itu dilakukan deklarasi sekaligus inisialisasi variabel *learning rate* (*alpha*) dengan nilai 0.6. variabel *learning rate* juga akan terus berubah sesuai dengan iterasi pelatihan.

12. Cek Epoch (Epoch < 100)

Pada tahap ini dilakukan pengecekan terhadap iterasi (*epoch*). Jika iterasi belum mencapai 100 maka proses latihan tetap dilakukan (ulangi langkah ke 13 sampai 18). Jika iterasi sudah mencapai 100 maka proses latihan dihentikan (lanjut ke langkah 19).

13. Perulangan Semua Pola Masukan

Pada tahap ini dilakukan proses pelatihan data. Data masukan berupa vektor *array* yang berisi jumlah warna hitam dan jumlah warna putih dari pola masukan (citra yang berisi sebuah obyek) yang sedang dilakukan pelatihan. Proses pelatihan data ini diulang sebanyak jumlah citra yang sudah dikenali sebagai obyek.

14. Hitung Jarak

Pada tahap ini dilakukan penghitungan jarak masukan terhadap bobot. Hasil keluaran tahap ini berupa nilai jarak (antara neuron masukan terhadap neuron keluaran) pada setiap neuron masukan sebanyak jumlah neuron keluaran. Setelah terhitung semua jarak maka dilakukan pencarian jarak terdekat (minimum). Keluaran pada tahap ini berupa indeks neuron keluaran yang memiliki jarak terdekat.

15. *Update Vektor Bobot*

Pada tahap ini masukan berupa indeks neuron keluaran dengan jarak terdekat. Neuron keluaran dengan jarak terdekat akan dilakukan pembaruan (*update*) bobot sesuai dengan jarak lama dan nilai *alpha (learning rate)*. Dengan demikian semua jarak akan berubah semakin mendekat ke neuron keluaran.

16. *Update Vektor Bobot Bias*

Pada tahap ini dilakukan pembaruan (*update*) bobot bias sesuai dengan neuron keluaran pemenang (neuron keluaran pemenang = 1, neuron keluaran bukan pemenang = 0) dan nilai *alpha (learning rate)*. Dengan demikian semua bobot bias akan berubah sesuai dengan neuron keluaran pemenang.

17. *Ambil Kelas Huruf*

Pada tahap ini neuron keluaran pemenang merupakan hasil pengelompokan (*clustering*) dari jenis huruf. Setiap neuron masukan akan memiliki kelas / kelompok sesuai dengan jarak terdekat terhadap neuron keluaran. Dengan demikian neuron keluaran pemenang merupakan jenis huruf yang dikenali.

18. *Pengubahan Epoch dan Alpha (Learning Rate)*

Pada tahap ini dilakukan pengubahan (*update*) nilai iterasi (*epoch*) dan nilai *learning rate (alpha)*. Nilai iterasi diubah dengan ketentuan nilai iterasi + 1 sedangkan nilai *learning rate* diubah dengan ketentuan nilai *learning rate* dibagi 2 (dikali 0.5).

19. *Hasil Clustering*

Pada tahap ini setiap neuron masukan sudah memiliki kelas sesuai dengan data pelatihan dan proses pelatihan. Hasil pengelompokan (*clustering*) ini sudah mencapai nilai yang konvergen (iterasi sebanyak 100 kali). Keluaran dari tahap ini berupa huruf yang sudah berbentuk *string*.

4. HASIL DAN PEMBAHASAN

Hasil implementasi SOM dalam OCR berbasis Android yaitu sebagai berikut.

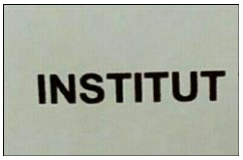
4.1. Akuisis Citra

Hasil pengambilan gambar dari implementasi akuisisi citra dapat dilihat pada Gambar 9.

Gambar 9.Hasil implementasi akuisisi citra

4.2. Smoothing

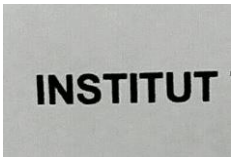
Hasil pengambilan gambar dari implementasi *smoothing* dapat dilihat pada Gambar 10.



Gambar 10. Hasil implementasi smoothing

4.3. Grayscaling

Hasil pengambilan gambar dari implementasi *grayscaling* dapat dilihat pada Gambar 11.



Gambar 11. Hasil implementasi grayscaling

4.4. Thresholding

Hasil pengambilan gambar dari implementasi *thresholding* dapat dilihat pada Gambar 12.

Gambar 12. Hasil implementasi thresholding

4.5. Scaling

Hasil pengambilan gambar dari implementasi *scaling* dapat dilihat pada Gambar 13.

Gambar 13. Hasil implementasi scaling

4.6. Connected Component Labeling

Hasil pengambilan gambar dari implementasi *segmentation connected component labeling* dapat dilihat pada Gambar 14.

Gambar 14(a). Hasil implementasi segmentation huruf pertama

Gambar 14(b).Hasil implementasi segmentation huruf kedua

Gambar 14(c).Hasil implementasi segmentation huruf ketiga

Gambar 14(d).Hasil implementasi segmentation huruf keempat

Gambar 14(e).Hasil implementasi segmentation huruf kelima

Gambar 14(f).Hasil implementasi segmentation huruf keenam

Gambar 14(g).Hasil implementasi segmentation huruf ketujuh

Gambar 14(h).Hasil implementasi segmentation huruf kedelapan

4.7. Sorting

Hasil pengambilan gambar dari implementasi *sorting* dapat dilihat pada Gambar 15.

Gambar 15(a).Hasil implementasi sorting huruf pertama

Gambar 15(b).Hasil implementasi sorting huruf kedua

Gambar 15(c).Hasil implementasi sorting huruf ketiga

Gambar 15(d).Hasil implementasi sorting huruf keempat

Gambar 15(e).Hasil implementasi sorting huruf kelima

Gambar 15(f).Hasil implementasi sorting huruf keenam

Gambar 15(g).Hasil implementasi sorting huruf ketujuh

Gambar 15(h).Hasil implementasi sorting huruf kedelapan

4.8. Pola Masukan (Ekstraksi Fitur)

Hasil pengambilan data dari implementasi pola masukan (ekstraksi fitur) dapat dilihat pada Tabel 1.

Tabel 1. Pola masukan (ekstraksi fitur)

No	Citra	Data / Pola
1		{1, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 5, 7, 8, 8, 9, 8, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
2		{4, 5, 5, 5, 5, 5, 6, 5, 6, 6, 6, 5, 5, 5, 4, 6, 6, 5, 7, 5, 5, 4, 4, 5, 4, 4, 9, 9, 9, 8, 8, 5, 3, 3, 5, 3, 4, 3, 2, 3, 3, 4, 9, 9, 9, 7, 8, 2, 0, 0}
3		{2, 4, 5, 6, 6, 5, 4, 4, 2, 2, 5, 6, 6, 4, 4, 4, 3, 5, 4, 3, 4, 6, 6, 6, 4, 0, 1, 5, 7, 8, 8, 5, 5, 5, 5, 4, 4, 5, 5, 5, 7, 7, 7, 7, 6, 2, 0, 0, 0, 0}
4		{0, 7, 8, 8, 8, 4, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 0, 1, 1, 1, 1, 1, 1, 7, 9, 9, 8, 8, 8, 2, 1, 1, 1, 2, 2, 1, 0, 0, 0, 0, 0}
5		{1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 9, 9, 9, 9, 9, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
6		{8, 8, 8, 8, 6, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 9, 9, 9, 9, 9, 9, 3, 2, 1, 1, 1, 2, 1, 0, 0, 0, 0}
7		{1, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 5, 5, 5, 5, 7, 8, 8, 8, 8, 9, 1, 2, 1, 1, 1, 1, 1, 1, 1, 8, 9, 8, 8, 8, 6, 0, 0, 0, 0}
8		{7, 7, 7, 8, 7, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 0, 2, 1, 1, 2, 2, 2, 2, 7, 9, 9, 9, 9, 9, 2, 2, 2, 2, 2, 1, 1, 0, 0, 0, 0}

4.9. Inisialisasi Vektor Bobot

Hasil inisialisasi bobot dapat dilihat sebagai pada Tabel 2 (perhitungan yang ditampilkan hanya *sample* 2 vektor bobot dari 26 vektor bobot).

Tabel 2. Inisialisasi vektor bobot

Vektor Bobot ke 1	Vektor Bobot ke 2
$W_{1\,1}$ = 0.380743819552	$W_{2\,1}$ = 0.0595971431457
$W_{1\,2}$ = 0.087477252131	$W_{2\,2}$ = 0.1101182491042
…	…
$W_{1\,50}$ = 0.011176155873	$W_{2\,50}$ = 0.2970232354200

4.10. Inisialisasi Vektor Bobot Bias

Nilai bobot bias pada setiap neuron keluarannya (26 neuron) adalah sebagai berikut.

$$b_i = e^{\left(1-\ln\left(\frac{1}{26}\right)\right)}$$

$$b_i = 70.67532753993514$$

4.11. Hitung Jarak

Hasil perhitungan jarak antar masukan terhadap bobot dapat dilihat pada Tabel 3 (perhitungan yang ditampilkan hanya *sample* 2 neuron masukan dari 79 neuron masukan).

Tabel 3. Hitung jarak

Neuron Masukan pertama (A1)	Neuron masukan kedua (B1)
$D_1 = (0.38074 - 3)^2 + (0.08747 - 4)^2 + \ldots + (0.01117 - 2)^2$ $D_1 = 1183.42873$	$D_1 = (0.38074 - 6)^2 + (0.08747 - 7)^2 + \ldots + (0.01117 - 0)^2$ $D_1 = 1711.07177$
$D_2 = 1172.57794$	$D_2 = 1694.25331$
…	…
$D_{10} = 1137.38109$	$D_{10} = 688.84800$
…	…
$D_{26} = 1189.45454$	$D_{26} = 1688.79779$

Setelah semua jarak antar masukan terhadap bobot (D_i) dengan setiap neuron masukan dihitung, selanjutnya mencari jarak terdekat dengan mencari hasil perhitungan terkecil. Pada perhitungan neuron masukan pertama (A1), neuron pemenang (jarak terdekat) adalah D_{10}. Sedangkan pada perhitungan neuron masukan kedua (B1), neuron pemenang adalah D_{10}. Dari hasil perhitungan tersebut maka neuron masukan pertama dan neuron masukan kedua termasuk ke dalam kategori (kelas) yang sama untuk iterasi pertama yaitu kategori 10.

4.12. Hitung Nilai Jarak

Hasil perhitungan nilai jarak dapat dilihat pada Tabel 4 (perhitungan yang ditampilkan hanya *sample* 2 neuron dari 79 neuron masukan).

Tabel 4. Hitung nilai jarak

Neuron Masukan pertama (A1)	Neuron masukan kedua (B1)
$a_1 = - 1183.42873 + 70.67532$ $a_1 = -1112.75340$	$a_1 = - 1711.07177 + 176.68831$ $a_1 = -1534.38345$
$a_2 = -1101.90261$	$a_2 = -1517.56499$
…	…
$a_{10} = -1066.70577$	$a_{10} = -684.43079$
…	…
$a_{26} = -1118.77921$	$a_{26} = -1512.10947$

Setelah semua nilai jarak (a_i) dihitung, selanjutnya mencari nilai jarak terbesar. Pada perhitungan neuron masukan pertama (A1), neuron pemenang (nilai terbesar) adalah a_{10}. Sedangkan pada perhitungan neuron masukan kedua (B1), neuron pemenang adalah a_{10}.

4.13. Pengaturan Keluaran Neuron

Hasil pengaturan keluaran neuron dapat dilihat pada Tabel 5 (pengaturan yang ditampilkan hanya *sample* 2 neuron dari 79 neuron masukan).

Tabel 5. Pengaturan keluaran neuron

Neuron Masukan pertama (A1)	Neuron masukan kedua (B1)
$y_1 = 0$	$y_1 = 0$
$y_2 = 0$	$y_2 = 0$
…	…
$y_{10} = 1$	$y_{10} = 1$
…	…
$y_{26} = 0$	$y_{26} = 0$

4.14. Update Vektor Bobot

Bobot yang diperbarui hanya bobot pada neuron masukan pemenang. Hasil pembaruan bobot dapat dilihat pada Tabel 6 (pembaruan bobot yang ditampilkan hanya *sample* 2 neuron dari 79 neuron masukan).

Tabel 6. Update vektor bobot

Neuron Masukan pertama (A1)	Neuron masukan kedua (B1)
$W_{10\,1} = 0.11527 + 0.6 (3 - 0.11527)$ $W_{10\,1} = 1.84611$	$W_{10\,1} = 1.84611 + 0.6 (6 - 1.84611)$ $W_{10\,1} = 4.33844$
$W_{10\,2} = 2.49566$	$W_{10\,2} = 5.19826$
…	…
$W_{10\,50} = 1.35017$	$W_{10\,50} = 0.54006$

4.15. Update Vektor Bobot Bias

Hasil pembaruan bobot bias dapat dilihat pada Tabel 7 (pembaruan bobot bias yang ditampilkan hanya *sample* 2 neuron dari 79 neuron masukan).

Tabel 7. Update vektor bobot bias

Neuron Masukan pertama (A1)	Neuron masukan kedua (B1)
$c_1 = (1 - 0.6)e^{(1-\ln(70.67532))} + 0.6 (0)$ $c_1 = 0.01538$ $b_1 = e^{(1-\ln(0.01538))}$ $b_1 = 176.68831$	$c_1 = (1 - 0.6)e^{(1-\ln(176.68831))} + 0.6 (0)$ $c_1 = 0.00615$ $b_1 = e^{(1-\ln(0.00615))}$ $b_1 = 441.72079$
$b_2 = 176.68831$	$b_2 = 441.72079$
....	...
$b_{10} = 4.41720$	$b_{10} = 3.21251$
....	...
$b_{26} = 176.68831$	$b_{26} = 441.72079$

4.16. Update Epoch dan Alpha (Learning Rate)

Hasil *update epoch* dan *alpha (learning rate)* dapat yaitu sebagai berikut.

Epoch = *Epoch* + 1

α = α x 0.5

α = 0.6 x 0.5

α = 0.3

4.17. Hasil Clustering

Hasil pengelompokan (*clustering*) dari pengujian SOM dengan iterasi sebanyak 100 kali dapat dilihat pada Tabel 8.

Tabel 8. Hasil pengelompokan (clustering)

Pola	Kelas pada iterasi ke -					
	10	**30**	**50**	**70**	**90**	**100**
A1	9	9	9	9	9	9
A2	9	9	9	9	9	9
A3	9	9	9	9	9	9
B1	7	7	7	7	7	7
B2	7	7	7	7	7	7
B3	7	7	7	7	7	7
C1	16	16	16	16	16	16
C2	16	16	16	16	16	16
C3	16	16	16	16	16	16
D1	7	7	7	7	7	7
D2	2	2	2	2	2	2
D3	2	2	2	2	2	2
E1	6	6	6	6	6	6
E2	6	6	6	6	6	6
E3	6	6	6	6	6	6
F1	19	19	19	19	19	19
F2	19	19	19	19	19	19
F3	19	19	19	19	19	19
G1	5	5	5	5	5	5
G2	18	18	18	18	18	18
G3	5	5	5	5	5	5
H1	1	1	1	1	1	1
H2	1	1	1	1	1	1
H3	1	1	1	1	1	1
I1	8	8	8	8	8	8
I2	8	8	8	8	8	8
I3	8	8	8	8	8	8
J1	11	11	11	11	11	11
J2	11	11	11	11	11	11
J3	11	11	11	11	11	11
K1	10	10	10	10	10	10
K2	10	10	10	10	10	10
K3	10	10	10	10	10	10
L1	13	13	13	13	13	13
L2	8	8	8	8	8	8
L3	13	13	13	13	13	13
M1	3	3	3	3	3	3
M2	3	3	3	3	3	3
M3	3	3	3	3	3	3
N1	1	1	1	1	1	1
N2	1	1	1	1	1	1
N3	1	1	1	1	1	1
O1	25	25	25	25	25	25
O2	2	2	2	2	2	2
O3	25	25	25	25	25	25
P1	21	21	21	21	21	21
P2	19	19	19	19	19	19
P3	21	21	21	21	21	21
Q1	5	5	5	5	5	5
Q2	5	5	5	5	5	5
Q3	5	5	5	5	5	5
R1	7	7	7	7	7	7
R2	7	7	7	7	7	7
R3	7	7	7	7	7	7
S1	14	14	14	14	14	14
S2	14	14	14	14	14	14
S3	14	14	14	14	14	14
T1	4	4	4	4	4	4
T2	4	4	4	4	4	4
T3	4	4	4	4	4	4
U1	0	0	0	0	0	0
U2	0	0	0	0	0	0
U3	0	0	0	0	0	0
V1	22	22	22	22	22	22
V2	22	22	22	22	22	22
V3	22	22	22	22	22	22
W1	24	24	24	24	24	24
W2	24	24	24	24	24	24
W3	24	24	24	24	24	24
X1	20	20	20	20	20	20
X2	20	20	20	20	20	20
X3	20	20	20	20	20	20
Y1	12	12	12	12	12	12
Y2	12	12	12	12	12	12
Y3	12	12	12	12	12	12
Z1	23	23	23	23	23	23
Z2	23	23	23	23	23	23
Z3	23	23	23	23	23	23
Input (I)	8	8	8	8	8	8

Dari Tabel 8 diperoleh informasi bahwa sebelum mencapai iterasi ke 10, hasil pengelompokan (clustering) sudah konvergen. Pola masukan (huruf I) dikenali sebagai huruf I karena masuk ke dalam kelas / kelompok / kategori ke 8. Walaupun terdapat pola huruf L2 yang juga termasuk ke dalam kategori ke 8 namun aplikasi menentukan hasil pengenalan huruf adalah huruf I karena aplikasi terakhir mendeteksi huruf dengan kategori ke 8 adalah pola huruf I3. Pengujian sudah dilakukan dan hasil untuk pola huruf pertama sudah didapat yaitu huruf I. Hasil akhir implementasi pengenalan karakter dapat dilihat pada Tabel 9.

Tabel 9. Hasil pengenalan

No	Masukan	Hasil
1		I
2		O
3		C
4		T
5		I
6		T
7		U
8		T

Hasil pengambilan gambar dari implementasi hasil pengenalan dapat dilihat pada Gambar 16.

Gambar 16.Implementasi menampilkan hasil pengenalan

Spesifikasi *smartphone* yang digunakan untuk melakukan pengujian dapat dilihat pada Tabel 10.

Tabel 10.Spesifikasi perangkat pengujian

Perangkat	Spesifikasi		
	***Smartphone* A**	***Smartphone* B**	***Smartphone* C**
Markah	Samsung Galaxy Note II N7100	Asus Zenfone™ Go Z00SD (ZC451TG)	Samsung Galaxy A5 (2016)
Processor	Exynos 4412 Quad-core 1.6 GHz	Mediatek MT6580 Quad-core 1.3 GHz	Qualcomm MSM8939 Snapdragon 615 Octa-core (4x1.2 GHz & 4x1.5 GHz)
Sistem Operasi	Android OS v4.4.2 (KitKat)	Android OS v5.1 (Lollipop)	Android OS v6.0.1 (Marshmallow)
Memori	2 GB RAM 16 GB Internal	1 GB RAM 8 GB Internal	2 GB RAM 16 GB Internal
Kamera	8 MP	5 MP	13 MP
Kerapatan Piksel	267 ppi	218 ppi	424 ppi

Hasil pengujian fungsional dapat dilihat pada Tabel 11.

Tabel 11.Pengujian fungsional

Pengujian	***Smartphone* A**		***Smartphone* B**		***Smartphone* C**	
	S	**G**	**S**	**G**	**S**	**G**
Layout *splash screen*						
Tampilan *splash screen*	√		√		√	
Layout utama						
Tampilan utama	√		√		√	
Masuk ke *open camera*	√		√		√	
Menampilkan gambar pada *imageview*	√		√			×
Masuk ke *image processing*	√		√			×
Layout *open camera*						
Tampilan *open camera*	√		√		√	
Akses kamera	√		√		√	
Ambil gambar	√		√		√	
Layout *croping*						
Tampilan *croping*	√		√		√	
Seleksi gambar	√		√		√	
Simpan gambar	√		√			×
Layout *image processing*						
Tampilan *image processing*	√		√			×
Menampilkan gambar pada *imageview*	√		√			×
Mengambil gambar dari memori	√		√			×
Smoothing	√		√			×
Grayscaling	√		√			×
Thresholding	√		√			×
Scaling	√		√			×
Segmentation (CCL)	√		√			×
Sorting	√		√			×
Pola masukan (ekstraksi fitur)	√		√			×
SOM	√		√			×
Menampilkan hasil pengenalan	√		√			×

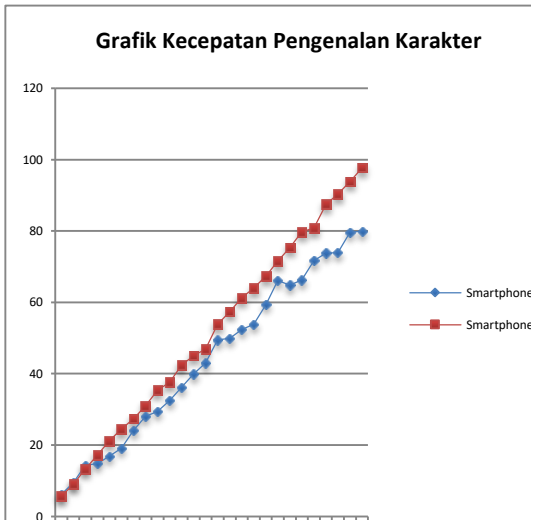
Keterangan :
S = Sukses G = Gagal
√ = Berjalan dengan baik
× = Tidak berjalan dengan baik

Hasil pengujian pengenalan karakter dapat dilihat pada Tabel 12.

Tabel 12.Hasil persentase pengenalan karakter

Kelompok pengujian	Kondisi pengujian	Jumlah huruf dikenali dengan benar pada *smartphone*			(%)
		A	B	C	
Huruf kapital (26 data)	Jarak 10 cm	5	18	-	44,23
	Jarak 20 cm	8	17	-	48,07
	Jarak 30 cm	12	15	-	51,92
	Jarak 40 cm	9	16	-	48,07
	Jarak 50 cm	9	14	-	44,23
	Cahaya *indoor* redup	6	16	-	42,3
	Cahaya *indoor* terang	10	16	-	50
	Cahaya *outdoor* tanpa sinar matahari	8	19	-	51,92
	Cahaya *outdoor* dengan sinar matahari	9	15	-	46,15
	Jenis *font* arial	12	15	-	51,92
	Jenis *font* century gothic	12	9	-	40,38
	Jenis *font* calibri	14	16	-	57,69
	Jenis *font* times new roman	7	3	-	19,23
	Kemiringan +15º	12	18	-	57,69
	Kemiringan +30º	8	11	-	36,53
	Kemiringan -15º	10	12	-	42,3
	Kemiringan -30º	5	10	-	28,84
Hasil persentase rata-rata					42,304

Hasil pengenalan karakter menggunakan metode SOM pada pengujian aplikasi berbasis Android menunjukkan tingkat keberhasilan pengenalan sebesar 42,304 %. Hasil pengujian kecepatan pengenalan karakter dapat dilihat pada Tabel 13 dan grafik kecepatan pengenalan karakter dapat dilihat pada Gambar 17.



Gambar 17.Grafik pengujian kecepatan pengenalan karakter

Tabel 13.Pengujian kecepatan pengenalan karakter

No	Karakter	Kecepatan (detik)			
		Smartphone A		*Smartphone* B	
		Lama	Rata-rata	Lama	Rata-rata
1	A	6,15	6,15	5,58	5,58
2	AB	9,45	4,72	9,03	4,51
3	ABC	14,36	4,78	13,3	4,43
4	ABCD	14,8	3,7	17,22	4,3
5	ABCDE	16,86	3,37	21,15	4,23
6	ABCDEF	19,13	3,18	24,45	4,07
7	ABCDEFG	24,13	3,44	27,37	3,91
8	ABCDEFGH	27,98	3,49	30,92	3,86
9	ABCDEFGHI	29,45	3,27	35,39	3,93
10	ABCDEFGHIJ	32,49	3,24	37,65	3,76
11	ABCDEFGHIJK	36,16	3,28	42,35	3,85
12	ABCDEFGHIJKL	39,92	3,32	45,06	3,75
13	ABCDEFGHIJKLM	42,96	3,3	46,9	3,6
14	ABCDEFGHIJKLMN	49,5	3,53	53,87	3,84
15	ABCDEFGHIJKLMNO	49,9	3,32	57,37	3,82
16	ABCDEFGHIJKLMNOP	52,39	3,27	61,28	3,83
17	ABCDEFGHIJKLMNOPQ	53,87	3,16	64,06	3,76
18	ABCDEFGHIJKLMNOPQR	59,34	3,29	67,4	3,74
19	ABCDEFGHIJKLMNOPQRS	66,1	3,47	71,59	3,76
20	ABCDEFGHIJKLMNOPQRST	64,86	3,24	75,32	3,76
21	ABCDEFGHIJKLMNOPQRSTU	66,28	3,15	79,74	3,79
22	ABCDEFGHIJKLMNOPQRSTUV	71,68	3,25	80,83	3,67
23	ABCDEFGHIJKLMNOPQRSTUVW	73,8	3,2	87,52	3,8
24	ABCDEFGHIJKLMNOPQRSTUVWX	73,88	3,07	90,4	3,76
25	ABCDEFGHIJKLMNOPQRSTUVWXY	79,67	3,18	93,87	3,75
26	ABCDEFGHIJKLMNOPQRSTUVWXYZ	79,95	3,07	97,75	3,75
Rata – rata		3,522		3,958	
Rata – rata keseluruhan		3,74			

Hasil pengujian kecepatan pengenalan karakter menunjukkan bahwa *smartphone* A memiliki kecepatan lebih tinggi daripada *smartphone* B dengan kecepatan rata-rata *smartphone* A sebesar 3.522 detik/karakter sedangkan pada *smartphone* B sebesar 3.958 detik/karakter. Rata-rata kecepatan pengenalan karakter dari seluruh perangkat pengujian yaitu 3.74 detik/karakter.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Kesimpulan dari hasil penelitian ini yaitu :

1. Berdasarkan hasil pengujian fungsional aplikasi terhadap beberapa *smartphone*, 15 dari 23 fungsi tidak dapat berjalan dengan baik dan benar pada *smartphone* C (Android v6.0.1 (Marshmallow)) dengan persentase keberhasilan 34.7 % namun berjalan dengan baik dan benar secara keseluruhan pada *smartphone* A (Android v4.2.2 (KitKat)) dan *smartphone* B (Android v5.1 (Lollipop)) dengan persentase keberhasilan 100 %.
2. Berdasarkan hasil pengujian pengenalan karakter dari beberapa jarak, cahaya, jenis *font* dan kemiringan diperoleh persentase rata – rata keberhasilan mengenali karakter dengan benar sebesar 42.304 %.
3. Berdasarkan hasil pengujian kecepatan pengenalan karakter, *smartphone* A (*processor* Exynos 4412 Quad-core 1.6 GHz dan 2 GB RAM) lebih cepat dibandingkan dengan *smartphone* B (*processor* Mediatek MT6580 Quad-core 1.3 GHz dan 1 GB RAM) dengan rata - rata kecepatan *smartphone* A 3.522 detik/karakter dan rata - rata kecepatan *smartphone* B 3.958 detik/karakter, dengan demikian diperoleh kecepatan rata - rata pengenalan karakter dari seluruh perangkat pengujian selama 3.74 detik/karakter.

5.2. Saran

Saran yang dapat diberikan setelah melakukan pengujian penelitian ini yaitu :

1. Pengenalan karakter dilakukan menggunakan metode lain atau menggunakan 2 metode (*hybrid*) supaya mendapat hasil yang lebih baik.
2. Menambahkan pengenalan pada karakter selain huruf kapital.
3. Dapat mengenali karakter dengan *style* miring (*italic*).
4. Menambahkan data *training* pada setiap jenis huruf supaya mendapatkan hasil yang lebih baik.
5. Menambahkan pengenalan pada latar belakang citra selain warna putih.
6. Dapat mengenali spasi (pengenalan kalimat).
7. Ditambahkannya fitur untuk memberi informasi bahwa citra yang sedang diamati tidak berisikan karakter jika citra tidak berisikan karakter.
8. Dapat mengenali karakter (kata atau kalimat) lebih dari satu baris.

DAFTAR PUSTAKA

[1] Anis, Y. and Isnanto, R.R., 2014. Penerapan Metode Self-Organizing Map (SOM) Untuk Visualisasi Data Geospasial Pada Informasi Sebaran Data Pemilih Tetap (DPT). JURNAL SISTEM INFORMASI BISNIS, 4(1), pp.48-57.

[2] Bradley, D. and Roth, G., 2007. Adaptive thresholding using the integral image. Journal of graphics, gpu, and game tools, 12(2), pp.13-21.

[3] Chandarana, J. and Kapadia, M., 2014. Optical character recognition. IJETAE Tranjact, 4(5).

[4] Huang, Z.K. and Chau, K.W., 2008. A new image thresholding method based on Gaussian mixture model. *Applied Mathematics and Computation*, *205*(2), pp.899-907.

[5] Lestari, W., 2010. Sistem Clustering Kecerdasan Majemuk Mahasiswa Menggunakan Algoritma Self Organizing Maps (SOM). Surakarta: STMIK Duta Bangsa.

[6] Pramesti, W.H.T., R.A., 2014. Pengenalan Karakter Teks Menggunakan Metode Neural Network Backpropagation. Jurnal Mahasiswa TEUB, 2(2).

[7] Murinto, M. and Agus, H., 2011. Segmentasi Citra Menggunakan Watershed Dan Itensitas Filtering Sebagai Pre Processing. Telematika, (6).

[8] Rindiani, H. and Nisa, K.K., 2015, June. Aplikasi Android untuk Pengenalan Citra Karakter Jepang dengan Library Tesseract. In *Makalah Seminar Ekstensi* (Vol. 2, No. 1).

[9] Sundari, S., 2012. Pembuatan Aplikasi LBS Bengkel Motor Resmi Menggunakan Eclipse Galileo Untuk Handphone Berbasis Android.

[10] Sutojo, T., Mulyanto, E., Suhartono, V. and Nurhayati, O.D., 2009. Teori Pengolahan Citra Digital.