PENGENALAN KARAKTER HURUF HANGUL KOREA MENGGUNAKAN RANDOM FOREST

Abdullah Imaduddin
abdimaduddin@gmail.com

Tjokorda Agung Budi W. ST., MT.
cokagung2001@gmail.com

Abstrak

Seiring berkembangnya teknologi informasi, rasa keingintahuan masyarakat terhadap budaya dan bahasa dari negara lain meningkat. Negara Korea adalah salah satu negara yang kebudayaannya sedang banyak diminati. Bahasa Korea ditulis menggunakan huruf hangul. *Optical character recognition* (OCR) adalah salah satu solusi untuk mempermudah dalam pengenalan karakter huruf hangul. Berbagai metode seperti ANN dan SVM umum digunakan pada OCR, namun keduanya memerlukan waktu training yang lama.

Random Forest digunakan sebagai metode alternatif dalam pengenalan karakter huruf Hangul Korea pada Tugas Akhir ini. Random Forest dapat menerima berbagai jenis input data dan menghasilkan nilai akurasi yang bagus. Hasil pengujian random forest dengan 10-tree dengan esktraksi ciri *projection based* mampu mengklarifikasi silabel huruf hangul berdasarkan KS5602 hingga 99%.

Kata Kunci : Optical Optical Character Recognition (OCR), pengenalan huruf Hangul, Random Forest, Projection Based Feature Extraction

I. Pendahuluan

Seiring dengan perkembangan teknologi dan informasi yang pesat diseluruh penjuru dunia, rasa keingintahuan masyarakat terhadap budaya dan bahasa dari negara lainpun ikut meningkat. Banyak negara yang dalam penulisan huruf bahasanya tidak menggunakan huruf romawi, melainkan menggunakan bentuk huruf lainnya. Adanya perbedaan penulisan huruf ini mempersulit masyarakat luar dalam proses pembelajaran budaya dan bahasa negara tersebut. Agar masyarakat luar dapat mempelajari budaya negara yang memiliki sistem penulisan huruf yang berbeda, maka masyarakat terpaksa memiliki dan memahami pengetahuan tata bahasa lokal dengan baik. Salah satu negara yang menggunakan sistem penulisan yang berbeda adalah negara Korea.

Negara Korea, terutama Korea Selatan, memiliki daya tarik yang kuat pada beberapa tahun terakhir. *Korean Wave* membuat masyarakat asing tertarik untuk mengenali budaya Korea lebih mendalam. Pengetahuan budaya Korea Selatan dapat ditemukan pada berbagai macam literatur, termasuk tulisan-tulisan yang terdapat pada media Internet. Tidak terlepas keharusan membaca literatur dalam bahasa Korea jika masyarakat ingin mengenali budaya Korea secara lebih mendalam. Oleh karenanya, masyarakat asing tentunya diharuskan memiliki kemampuan untuk dapat membaca dan memahami literatur yang tertulis dalam

hangul guna mempelajari budaya Korea Selatan secara lebih mendalam.

Google Translate adalah sebuah alat bantu yang dapat mengartikan tulisan hangul Korea yang tertulis dalam media elektronik kedalam bahasa lain, seperti bahasa Indonesia ataupun bahasa Inggris. Kendala yang dihadapi adalah ketika suatu literatur yang ingin dibaca tidaklah tertulis pada media elektronik, melainkan pada media cetak, seperti buku, surat kabar, dan majalah.

Pengenalan Karakter Optik atau lebih dikenal dengan *Optical Character Recognition* (OCR) merupakan sebuah solusi yang dapat digunakan dalam mengenali karakter huruf dari sebuah gambar yang kemudian dikeluarkan kembali dalam bentuk teks[2]. Banyak penelitian OCR yang sudah dilakukan dalam pengenalan huruf Hangul, yang kemudian diterapkan menjadi sebuah aplikasi. Tingkat pengenalan rata-rata pada berbagai penelitian terkait sudah sangat tinggi. Meski demikian, masih terdapat kesalahan yang ditemui dalam pengenalan huruf Hangul Korea. Kesalahan pengenalan pada umumnya disebabkan oleh banyaknya susunan kombinasi huruf Hangul yang tersedia yang mengakibatkan sulitnya proses segmentasi huruf. Jumlah suku kata yang dapat dibentuk dari kombinasi huruf hangul mencapai 11172 karakter, namun hanya 2350 suku kata yang digunakan pada penulisan sehari-hari[7]. Metode yang umum digunakan pada pengenalan huruf hangul adalah metode *template matching*[2][8].

II. Dasar teori

A. *Edge Detection*

Deteksi tepi adalah nama untuk satu set metode matematika yang bertujuan untuk mengidentifikasi titik-titik dalam gambar digital di mana kecerahan gambar perubahan tajam atau, dalam kata lain, memiliki diskontinuitas. Titik-titik di mana kecerahan gambar mengalami perubahan yang tajam biasanya diatur dalam satu set segmen garis melengkung disebut tepi. Masalah yang sama untuk menemukan diskontinuitas pada sinyal 1D dikenal sebagai deteksi langkah dan masalah menemukan diskontinuitas sinyal dari waktu ke waktu dikenal sebagai deteksi perubahan. Deteksi tepi adalah alat fundamental dalam pengolahan citra, visi mesin dan visi komputer, khususnya di bidang fitur deteksi dan ekstraksi fitur[11].

Metode *Edge Detection* yang paling kuat yang sering digunakan adalah metode Canny. Metode Canny berbeda dari metode pendeteksian tepi lain karena menggunakan dua ambang batas yang berbeda (untuk mendeteksi tepi kuat dan lemah). Metode Canny memiliki sebuah *treshold* yang memisahkan antara garis tepi lemah dan garis tepi kuat. Setelah garis tepi kuat dan garis tepi lemah terdeteksi, metode akan menelusuri ulang garis-garis yang terdeteksi dengan ambang batas lemah. Jika garis tersebut terhubung dengan garis yang terdeteksi dengan ambang batas kuat, maka garis-garis lemah tersebut akan dimasukkan kedalam *output* akhir. Oleh karena itu, metode ini lebih rentan terhadap *noise* dibandingkan dengan metode Edge Detection yang lain, dan lebih mungkin untuk mendeteksi tepi lemah dengan benar.

B. *Reverse Edge Detection*

Metode *reverse edge detection*, pada dasarnya memiliki alur proses yang sama dengan metode *edge detection*. Tujuan penggunaan metode ini adalah untuk mengurangi *noise* yang terdapat pada gambar. *Salt & pepper* adalah salah satu jenis *noise* yang biasa ditemukan pada suatu gambar. Posisi pixel-pixel *noise* tersebut tersebar pada seluruh permukaan gambar.

Pada *reverse edge detection*, program akan mendeteksi nilai pixel hitam mulai dari kiri atas sampai kanan bawah. Ketika ditemukan pixel berwarna hitam,

maka akan dilakukan pengecekan terhadap pixel-pixel tetangganya. Jika ditemukan adanya pixel hitam yang berketetanggaan, maka pixel tersebut tetap dimasukkan kedalam gambar akhir. Namun, ketika tidak ditemukan pixel hitam yang berketetanggaan, maka pixel tersebut akan dieliminasi dari gambar akhir.

C. *Projection Based Feature Extraction*

Projection histograms diperkenalkan pada tahun 1956 oleh Glauberman dalam sistem hardware OCR. Metode ini bekerja dengan melakukan perhitungan sederhana terhadap pixel-pixel hitam yang berada pada suatu baris. Baris yang digunakan dapat berupa horizontal ataupun vertical. Belakangan ini, teknik ini banyak digunakan untuk segmentasi baris, kata, dan karakter[10].

Untuk setiap baris dan kolom pada gambar, akan dihitung jumlah pixel hitam yang terdapat pada baris dan kolom tersebut. Pada penelitian ini, peneliti akan menyimpan data histogram gambar yang berukuran 30x30 pixel dalam bentuk vektor dengan ukuran 1x60. Dimana kolom 1-30 berisikan data jumlah pixel hitam untuk baris 1-30 dan kolom 31-60 berisikan data jumlah pixel hitam untuk kolom 1-30.

D. *Decision Tree*

Decision Tree atau Pohon Keputusan adalah sebuah metode learning yang menggunakan struktur pohon (tree), dimana informasi mengenai prediksi yang dilakukan tersimpan pada setiap node-leaf tree. Pohon keputusan dibangun menggunakan proses pelatihan data pada setiap node. Algoritma learning pada decision tree akan memilih salah satu atribut dari dataset yang telah memenuhi kriteria tertentu. Node turunan akan dibuat dengan memecah sampel data training yang telah ada berdasarkan nilai atribut yang telah ditentukan. Proses ini akan terus berulang sampai suatu kondisi terpenuhi, atau sebanyak jumlah yang sudah ditentukan, misal sebanyak x kali atau sebanyak jumlah data sampel yang tersedia[3].

Pada decision tree klasifikasi dilakukan dengan menelusuri node mulai dari root sampai ke node-leaf, sesuai dengan kondisi atribut pada tiap node. Selain untuk klasifikasi, decision tree juga memiliki varian untuk memecahkan masalah regresi, yang biasa disebut Regression Tree.

E. *Random Forest*

Random forest adalah algoritma klasifikasi yang menggunakan ensemble learning. Random forest dikembangkan oleh Leo Breiman dan Adele Cutler. Random forest didasarkan pada sebuah ide untuk membentuk suatu kumpulan dari decision tree dengan variansi yang dapat diatur[1].

Ensemble adalah pendekatan *divide and conquer* yang digunakan untuk meningkatkan kinerja. Prinsip utama di balik metode ensemble adalah bahwa kelompok *"weak-learner"* dapat dikumpulkan dan membentuk sebuah *"strong-learner"*. *Runtimes Random Forest* cukup cepat, dan mampu menangani data yang tidak seimbang dan tidak lengkap. Kelemahan Random Forest pada regresi tidak dapat memprediksi nilai yang diluar jangkauan pada data training., dan memiliki kemungkinan melakukan *over-fit* pada data yang memiliki sangat banyak *noise.*

Tidak seperti decision tree yang cenderung sulit diimplementasikan pada data dengan variansi yang tinggi, random forest memberikan nilai rata-rata untuk menemukan titik *balance* pada data-data

tersebut. Random Forest tahan terhadap noise yang terdapat pada data.

Algoritma *training* untuk Random Forest adalah dengan menggunakan Bootstrap Aggregating (Bagging). Proses latih dilakukan dengan mengambil satu set data latih yang kemudian akan dimasukkan kedalam suatu tree. Pemilihan atribut dalam setiap kali sebuah node akan dipecah diambil secara acak. Bagging melakukan pemilihan *sample* berulang kali, dengan penggantian. Jumlah data latih yang diberikan pada setiap pohon akan berjumlah sama. Kolom data yang digunakan pada suatu node akan ditentukan nilai *treshold* nya menggunakan *gini index*[5].

$$\text{Gini} = 1 - \sum_{i=1}^{m} P_i^2$$

$$\text{Gini}_{split} = \frac{|D_1|}{|D|}\text{Gini}(D_1) + \frac{|D_2|}{|D|}\text{Gini}(D_2)$$

m adalah jumlah kelas, sedangkan Pi adalah rasio jumlah data yang diberi label kelas i dalam D. Gini index menghasilkan *binary split* untuk setiap atribut. Perhitungan nilai Gini *index binary split* dimulai dengan membagi data D menjadi 2 kelompok data D1 dan D2. Nilai Gini D dihitung dengan menjumlahkan nilai Gini setiap partisi yang sudah diberi bobot. Untuk nilai Gini diskrit, setiap subset nilai dari atribuat akan dipertimbangkan untuk dijadikan nilai split pada suatu label. Split yang menghasilkan nilai Gini terkecil akan dipilih sebagai split *treshold* sebuah node.

III. Perancangan Sistem

A. Skenario Pengujian

Pengujian penelitian ini dilakukan dengan 7050 data latih dan 25850 data uji yang akan digunakan tergantung pada skenario pengujian. Pengujian sistem dibagi menjadi 3 skenario yang kemudian akan direpresentasikan sesuai metode yang digunakan pada pembentukan random forest Pengujian sistem dibagi menjadi 3 skenario, yaitu:

1. Pengujian Akurasi Sistem Terhadap Data Latih
2. Pengujian Akurasi Pengenalan Sistem Terhadap Data Uji dengan Noise
3. Pengujian Akurasi Pengenalan Sistem Terhadap Data Uji dengan Font Asing

B. Hasil Pengujian

Hasil dari pengujian skenario pertama adalah nilai akurasi dari masing-masing pengujian, yang dijelaskan pada Tabel 1 dan Tabel 2 dibawah ini.

Tabel 1: Nilai Akurasi Berdasarkan Pengujian Skenario I dengan 5-tree

Uji	Metode		Dikenali	Akurasi
1	-	Vectorization	2219	94.43%
2	-	Vectorization	2188	93.11%
3	-	Vectorization	2190	93.19%
4	-	Projection Based	2209	94.00%
5	-	Projection Based	2211	94.09%
6	-	Projection Based	2173	92.47%
7	Edge Detection	Vectorization	2188	93.11%
8	Edge Detection	Vectorization	2187	93.06%
9	Edge Detection	Vectorization	2143	91.19%
10	Edge Detection	Projection Based	2204	93.79%
11	Edge Detection	Projection Based	2208	93.96%
12	Edge Detection	Projection Based	2183	92.89%
13	Reverse Edge Detection	Vectorization	978	41.62%
14	Reverse Edge Detection	Vectorization	383	16.30%
15	Reverse Edge Detection	Vectorization	307	13.06%
16	Reverse	Projection	2164	92.09%

	Edge Detection	Based		
17	Reverse Edge Detection	Projection Based	2180	92.77%
18	Reverse Edge Detection	Projection Based	2143	91.19%

Tabel 2: Hasil Pengujian Skenario I dengan 10-tree

Uji	Metode		Dikenali	Akurasi
1	-	Vectorization	2340	99.57%
2	-	Vectorization	2324	98.89%
3	-	Vectorization	2299	97.83%
4	-	Projection Based	2338	99.49%
5	-	Projection Based	2327	99.02%
6	-	Projection Based	2322	98.81%
7	Edge Detection	Vectorization	2334	99.32%
8	Edge Detection	Vectorization	2337	99.45%
9	Edge Detection	Vectorization	2312	98.38%
10	Edge Detection	Projection Based	2345	99.79%
11	Edge Detection	Projection Based	2341	99.62%
12	Edge Detection	Projection Based	2326	98.98%
13	Reverse Edge Detection	Vectorization	1352	57.53%
14	Reverse Edge Detection	Vectorization	486	20.68%
15	Reverse Edge Detection	Vectorization	410	17.45%
16	Reverse Edge Detection	Projection Based	2333	99.28%
17	Reverse Edge Detection	Projection Based	2328	99.06%
18	Reverse Edge Detection	Projection Based	2318	98.64%

Berdasarkan hasil pengujian yang dilakukan pada skenario I, dapat disimpulkan bahwa tidak semua data uji dapat dikenali walaupun data yang diujikan adalah data yang sama dengan data latih. Hal ini diduga terjadi karena adanya ciri identik yang dimiliki oleh *class* lain, yang mana membuat prediksi random forest tidak tepat

Hasil dari pengujian skenario pertama adalah nilai akurasi dari masing-masing pengujian, yang dijelaskan pada Tabel 1 dan Tabel 3 sampai Tabel 6 dibawah ini.

Tabel 3: Nilai Akurasi Berdasarkan Pengujian Skenario II noise 5% 5-tree

Uji	Metode		Dikenali	Akurasi
1	-	Vectorization	1814	77.19%
2	-	Vectorization	1827	77.75%
3	-	Vectorization	1779	75.70%
4	-	Projection Based	967	41.15%
5	-	Projection Based	1153	49.06%
6	-	Projection Based	1024	43.57%
7	Edge Detection	Vectorization	1225	52.13%
8	Edge Detection	Vectorization	1395	59.36%
9	Edge Detection	Vectorization	1324	56.34%
10	Edge Detection	Projection Based	670	28.51%
11	Edge Detection	Projection Based	897	38.17%
12	Edge Detection	Projection Based	763	32.47%
13	Reverse Edge Detection	Vectorization	763	32.47%
14	Reverse Edge Detection	Vectorization	302	12.85%

15	Reverse Edge Detection	Vectorization	264	11.23%
16	Reverse Edge Detection	Projection Based	1507	64.13%
17	Reverse Edge Detection	Projection Based	1541	65.57%
18	Reverse Edge Detection	Projection Based	1385	58.94%

Tabel 4: Nilai Akurasi Berdasarkan Pengujian Skenario II noise 5% 10-tree

Uji	Metode		Dikenali	Akurasi
1	-	Vectorization	2150	91.49%
2	-	Vectorization	2124	90.38%
3	-	Vectorization	2102	89.45%
4	-	Projection Based	1504	64.00%
5	-	Projection Based	1686	71.75%
6	-	Projection Based	1562	66.47%
7	Edge Detection	Vectorization	1547	65.83%
8	Edge Detection	Vectorization	1977	84.13%
9	Edge Detection	Vectorization	1886	80.26%
10	Edge Detection	Projection Based	1173	49.92%
11	Edge Detection	Projection Based	1397	59.45%
12	Edge Detection	Projection Based	1267	53.92%
13	Reverse Edge Detection	Vectorization	1089	46.34%
14	Reverse Edge Detection	Vectorization	397	16.89%
15	Reverse Edge Detection	Vectorization	354	15.06%
16	Reverse Edge Detection	Projection Based	2018	85.87%
17	Reverse Edge Detection	Projection Based	2041	86.85%
18	Reverse Edge Detection	Projection Based	1925	81.92%

Tabel 5: Nilai Akurasi Berdasarkan Pengujian Skenario II noise 15% 10-tree

Uji	Metode		Dikenali	Akurasi
1	-	Vectorization	1009	42.94%
2	-	Vectorization	1058	45.02%
3	-	Vectorization	948	40.34%
4	-	Projection Based	256	10.89%
5	-	Projection Based	327	13.92%
6	-	Projection Based	282	12.00%
7	Edge Detection	Vectorization	476	20.26%
8	Edge Detection	Vectorization	707	30.09%
9	Edge Detection	Vectorization	640	27.23%
10	Edge Detection	Projection Based	313	13.32%
11	Edge Detection	Projection Based	394	16.77%
12	Edge Detection	Projection Based	332	14.13%
13	Reverse Edge Detection	Vectorization	480	20.43%
14	Reverse Edge Detection	Vectorization	214	9.11%
15	Reverse Edge Detection	Vectorization	172	7.32%
16	Reverse Edge Detection	Projection Based	689	29.32%
17	Reverse Edge Detection	Projection Based	574	24.43%
18	Reverse	Projection	415	17.66%

	Edge Detection	Based		

Tabel 6: Nilai Akurasi Berdasarkan Pengujian Skenario II noise 15% 10-tree

Uji	Metode		Dikenali	Akurasi
1	-	Vectorization	1245	52.98%
2	-	Vectorization	1234	52.51%
3	-	Vectorization	1119	47.62%
4	-	Projection Based	353	15.02%
5	-	Projection Based	568	24.17%
6	-	Projection Based	441	18.77%
7	Edge Detection	Vectorization	928	39.49%
8	Edge Detection	Vectorization	1076	45.79%
9	Edge Detection	Vectorization	1118	47.57%
10	Edge Detection	Projection Based	533	22.68%
11	Edge Detection	Projection Based	712	30.30%
12	Edge Detection	Projection Based	602	25.62%
13	Reverse Edge Detection	Vectorization	744	31.66%
14	Reverse Edge Detection	Vectorization	264	11.23%
15	Reverse Edge Detection	Vectorization	246	10.47%
16	Reverse Edge Detection	Projection Based	1221	51.96%
17	Reverse Edge Detection	Projection Based	997	42.43%
18	Reverse Edge Detection	Projection Based	850	36.17%

Berdasarkan hasil pengujian dengan skenario II menggunakan data uji yang diberi *noise salt pepper* dengan tingkat *noise* yang berbeda, dapat disimpulkan bahwa semakin besar tingkat *noise* yang diberikan, semakin kecil nilai akurasi pengenalan. Semakin besarnya tingkat *noise* maka semakin besar perubahan nilai data yang terekstrak pada tahap ekstraksi ciri. Sehingga mempengaruhi hasil akhir prediksi *class* pada random forest

Hasil dari pengujian skenario pertama adalah nilai akurasi dan kecepatan proses dari masing-masing tahap, yang dijelaskan pada Tabel 7 dan Tabel 8 dibawah ini.

Tabel 7: Nilai Akurasi Berdasarkan Pengujian Skenario III dengan Font Asing

Uji	Data Uji	Metode		Dikenali	Akurasi
1	Gungsuh	-	Vectorization	14	0.60 %
2	Malgun	-	Vectorization	4	0.17 %
3	Gungsuh	-	Projection Based	18	0.77 %
4	Malgun	-	Projection Based	16	0.68 %
5	Gungsuh	Edge Detection	Vectorization	4	0.17 %
6	Malgun	Edge Detection	Vectorization	1	0.04 %
7	Gungsuh	Edge Detection	Projection Based	6	0.26 %
8	Malgun	Edge Detection	Projection Based	11	0.47 %
9	Gungsuh	Reverse Edge Detection	Vectorization	4	0.17 %
10	Malgun	Reverse Edge Detection	Vectorization	2	0.09 %
11	Gungsuh	Reverse Edge Detection	Projection Based	9	0.38 %

12	Malgun	Reverse Edge Detection	Projection Based	6	0.26 %

Tabel 8: Nilai Akurasi Berdasarkan Pengujian Skenario III dengan font Asing

Uji	Data Uji	Metode		Dikenali	Akurasi
1	Gungsuh	-	Vectorization	26	1.11 %
2	Malgun	-	Vectorization	4	0.17 %
3	Gungsuh	-	Projection Based	14	0.60 %
4	Malgun	-	Projection Based	15	0.64 %
5	Gungsuh	Edge Detection	Vectorization	4	0.17 %
6	Malgun	Edge Detection	Vectorization	3	0.13 %
7	Gungsuh	Edge Detection	Projection Based	6	0.26 %
8	Malgun	Edge Detection	Projection Based	12	0.51 %
9	Gungsuh	Reverse Edge Detection	Vectorization	3	0.13 %
10	Malgun	Reverse Edge Detection	Vectorization	2	0.09 %
11	Gungsuh	Reverse Edge Detection	Projection Based	12	0.51 %
12	Malgun	Reverse Edge Detection	Projection Based	6	0.26 %

Berdasarkan hasil pengujian dengan skenario III dapat disimpulkan bahwa nilai akurasi pengenalan data uji yang merupakan data asing sangat rendah. Rata-rata akurasi pengujian masih di bawah 1%. Nilai akurasi yang rendah disebabkan oleh tidak adanya klasifikasi yang dilakukan sebelumnya terhadap font-font tersebut.

Analisa hasil pengujian dan perbandingan grafik dapat disimpulkan sebagai berikut:

a. Pada kasus pengujian skenario I menghasilkan nilai akurasi yang tinggi, dengan akurasi lebih besar dari 90% untuk 5 metode, dan di bawah 90% untuk metode *reverse edge detection* dengan *vectorization.*
b. Pada kasus pengujian skenario II dengan *noise* 5% menghasilkan nilai akurasi yang beragam, mulai dari 11% sampai 77% dan 15% sampai 91% untuk pengujian dengan menggunakan random forest 5-tree dan 10-tree.
c. Pada kasus pengujian skenario II dengan *noise* 15% menghasilkan nilai akurasi yang beragam, mulai dari 7% sampai 45% dan 10% sampai 52% untuk pengujian dengan menggunakan random forest 5-tree dan 10-tree.
d. Pada kasus pengujian skenario III dengan menggunakan data uji berupa font asing didapatkan hasil akurasi 0.04% sampai 0.77% dan 0.09% sampai 1.11% untuk pengujian dengan menggunakan random forest 5-tree dan 10-tree.
e. Dari 6 kombinasi metode yang digunakan, metode *no-preprocessing* dengan *vectorization* menghasilkan rata-rata nilai akurasi yang lebih tinggi dibandingkan dengan 5 metode lainnya. Tingginya akurasi pada metode ini disebabkan oleh tidak adanya *preprocessing* yang tidak merubah data gambar serta *vectorization* yang menyimpan ciri lebih banyak dibandingkan dengan metode *projection based.*
f. Dari 6 kombinasi metode yang digunakan, metode *reverse edge detection* dengan *vectorization*

menghasilkan nilai akurasi yang paling rendah dibandingkan dengan 5 metode lainnya. Rendahnya nilai akurasi disebabkan oleh banyaknya data gambar yang tereliminasi pada tahap *preprocessing* menggunakan *reverse edge detection*.

IV. Kesimpulan

1. Random Forest masih dapat memberikan hasil pengenalan yang tidak tepat meskipun data yang diujikan adalah data yang digunakan pada saat training.
2. Pengujian dengan data uji yang adalah data latih yang diberi noise menghasilkan nilai akurasi yang cukup rendah.
3. Pengujian dengan data uji yang menggunakan jenis font yang tidak digunakan pada proses training menghasilkan nilai akurasi yang sangat rendah.
4. Metode no-preprocessing dengan vectorization menghasilkan nilai akurasi yang paling tinggi dibandingkan dengan 5 metode lain yang digunakan. Data gambar tidak berkurang karena tidak adanya tahap preprocessing dan ciri yang terekstraksi lebih banyak.
5. Metode reverse edge detection dengan vectorization menghasilkan nilai akurasi paling rendah dibandingkan dengan 5 metode lain yang digunakan. Banyak data gambar tereliminasi dengan menggunakan reverse edge detecetion dan dengan sedikitnya data gambar, metode vectorization tidak cocok digunakan karena terlalu banyaknya sebaran data pada satu vektor.
6. Pada pengujian dengan menggunakan metode yang sama, pengujian yang menggunakan random forest dengan menggunakan 10-tree menghasilkan nilai akurasi yang lebih besar dibandingkan dengan random forest menggunakan 5-tree.

.

V. Daftar pustaka

[1] Breiman, Leo. 2001. Random Forests. Kluwer-Academic Publishers.

[2] Cheriet, M. et al. 2007. Character Recognition Systems: A Guide for Students and Practitioners. John Wiley & Sons, Inc.

[3] Criminisi, A., Shotton, J. 2013. Decision Forest for Computer Vision and Medical Image Analysis. Springer-Verlag London.

[4] Guyon, I., Elisseeff, A. An Introduction to Variable and Feature Selection.

[5] Han, Jiawei., Kamber, Micheline., dan Pei, Jian. 2012. Data Mining Concepts and Techniques. Elsevier Inc.

[6] Kim, J.W. The Korean Tradition of Translation: From the Primeval Period to the Modern Era.

[7] Lee, J.S., Kwon, O.J., Bang S.Y. Highly accurate recognition of printed Korean characters through an improved two-stage classification method.

[8] Liu, C.L., Fujisawa H. Classification and Learning for Character Recognition: Comparison of Methods and Remaining Problems.

[9] Mitchell, Tom. 1997, Machine Learning, McGraw-Hill.

[10] Naser, M.A., Hamid, N.I.B., Hoque, M.A. Projection Based Feature Extraction Process For Bangla Script: A Modified Approach.

[11] Nixon, M.S., Aguado, A.S. 2012. Feature Extraction & Image Processing for Computer Vision. United Kingdom. Elsevier Ltd.

[12] Parker, J.R. 2011. Algorithms for Image Processing and Computer Vision. Wiley Publishing, Inc.

[13] Random Forest. http://www.ualberta.ca/~drr3/random-forest.html diakses pada 11 agustus 2014.