PENGENALAN HANGUL MENGGUNAKAN *SUPPORT VECTOR MACHINE*

TESIS

PROGRAM MAGISTER TEKNIK ELEKTRO
MINAT SISTEM KOMUNIKASI DAN INFORMATIKA

Ditujukan untuk memenuhi persyaratan
memperoleh gelar Magister Teknik

Rahmatina Hidayati
NIM. 146060300111028

UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2018

TESIS

PENGENALAN HANGUL MENGGUNAKAN *SUPPORT VECTOR MACHINE*

RAHMATINA HIDAYATI
NIM. 146060300111028

Telah dipertahankan di depan penguji
pada Tanggal : **26 Juli 2018**
dinyatakan telah memenuhi syarat
untuk memperoleh gelar Magister Teknik

Komisi Pembimbing,

Pembimbing I	Pembimbing II
Dr. M. Sarosa, Dipl. Ing., M.T.	**Dr.Eng. Panca Mudjirahardjo, S.T., M.T.**

Malang,

Universitas Brawijaya
Fakultas Teknik, Jurusan Teknik Elektro
Ketua Program Magister **Teknik Elektro**

Dr.Eng. Panca Mudjirahardjo, S.T., M.T.
NIP. 197003292000121001

JUDUL TESIS :

PENGENALAN HANGUL MENGGUNAKAN *SUPPORT VECTOR MACHINE*

Nama Mahasiswa	:	Rahmatina Hidayati
NIM	:	146060300111028
Program Studi	:	Teknik Elektro
Kekhususan / Minat	:	Sistem Komunikasi dan Informatika
KOMISI PEMBIMBING	:	
Ketua	:	Dr. Moechammad Sarosa, Dipl.Ing., M.T.
Anggota	:	Dr. Eng. Panca Mudjirahardjo, S.T., M.T.
TIM DOSEN PENGUJI		
Dosen Penguji I	:	Dr. Ir. Erni Yudaningtyas, M.T.
Dosen Penguji II	:	Dr. Ir. Bambang Siswojo, M.T.
Tanggal Ujian	:	26 Juli 2018
SK Penguji	:	1575 tahun 2018

PERNYATAAN ORISINALITAS TESIS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan Saya dan berdasarkan hasil penelusuran berbagai karya ilmiah, gagasan dan masalah ilmiah yang diteliti dan diulas di dalam Naskah Tesis ini adalah asli dari pemikiran Saya. Tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah Tesis ini dapat dibuktikan terdapat unsur-unsur jiplakan, Saya bersedia Tesis dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 tahun 2003, Pasal 25 Ayat 2 dan Pasal 70).

Malang, 27 Juli 2018

Mahasiswa,

Nama : **RAHMATINA HIDAYATI**

NIM : **146060300111028**

PM : **TEKNIK ELEKTRO**

PROGRAM MAGISTER TEKNIK ELEKTRO

Karya ilmiah ini penulis tujukan kepada

bapak dan ebok tercinta.

RIWAYAT HIDUP

Rahmatina Hidayati, dilahirkan di Pamekasan 20 Februari 1989 dari pasangan Achyak dan Suhawati. Pendidikan SD, SMP, dan SMA diselesaikan di kota Pamekasan. Selepas SMA penulis melanjutkan studi di Universitas Trunojoyo Madura dengan jurusan Teknik Informatika. Pada tahun 2014 hingga 2018, penulis melanjutkan studi magister di Program Magister Teknik Elektro Universitas Brawijaya dengan minat Sistem Komunikasi dan Informatika.

UCAPAN TERIMA KASIH

Segala puji bagi Allah SWT atas segala karunia yang telah diberikan sehingga penulis dapat menyelesaikan karya ilmiah ini. Dalam pelaksanaan pembuatan tesis ini, penulis juga banyak menerima bantuan dari berbagai pihak. Oleh karena itu,dengan rasa hormat penulis mengucapkan terima kasih kepada pihak-pihak yang telah membantu antara lain:

1. Kedua orang tua dan keluarga besar penulis atas dukungan yang sudah diberikan.
2. Bapak Dr. M. Sarosa, Dipl.Ing., M.T., dan bapak Dr. Eng. Panca Mudjirahardjo, M.T. selaku pembimbing, terima kasih atas semua arahan dalam penyelesaian penelitian tesis ini.
3. Ibu Dr. Ir, Erni Yudaningtyas, M.T., dan bapak Ir. Bambang Siswojo, M.T., selaku dewan penguji, terima kasih atas saran dan masukannya yang membuat penelitian ini menjadi lebih detail.
4. Bapak Dr. Eng. Panca Mudjirahardjo, S.T., M.T. selaku Ketua Program Studi Magister Teknik Elektro, beserta seluruh dosen pengajar dan staf di Program Magister Teknik Elektro Universitas Brawijaya atas sistem pembelajaran dan ilmu pengetahuan yang sudah diberikan.
5. Seluruh rekan-rekan di SKI 2014 dan 2015.

Dan semua pihak yang tidak bisa disebutkan satu persatu,yang secara langsung maupun tidak langsung turut membantu menyelesaikan Tesis ini, terima kasih atas segala dukungan dan doa yang diberikan kepada penulis, semoga Allah SWT melimpahkan rahmat dan hidayat sebagai imbalan atas kebaikannya.

RINGKASAN

Rahmatina Hidayati, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Juli 2018, Pengenalan Hangul Menggunakan Support Vector Machine, Dosen Pembimbing: Moechammad Sarosa dan Panca Mudjirahardjo.

Pengenalan citra Hangul cenderung lebih sulit dibandingkan dengan pengenalan citra Latin. Hal ini bisa dilihat dari bagaimana menyusunnya. Hangul disusun dengan dua dimensi, sedangkan Latin dari kiri ke kanan. Dalam penelitian ini dibuat sebuah sistem untuk mengenali citra Hangul menjadi teks Latin yang diharapkan dapat digunakan sebagai bahan pembelajaran bagaimana membaca Hangul itu sendiri. Pada umumnya, sistem pengenalan citra terbagi ke dalam 3 langkah, pertama preprocessing yang meliputi tresholding, segmentasi yang dilakukan menggunakan metode connected component-labeling, dan thinning atau penipisan menerapkan algoritma Zhang-Suen. Kedua, sistem melakukan pengambilan fitur dari masing-masing citra. Metode yang diterapkan dalan pencarian fitur adalah chain code.Terakhir, proses pengenalan yang menerapkan Support Vector Machine (SVM) dengan beberapa kernel (linear, polynomial, radial basis function, dan laplacian). Dalam percobaan dilakukan pengenalan pada citra huruf dan kata Hangul. Untuk huruf terdiri dari 34 huruf, di mana masing-masing huruf memiliki 15 pola berbeda. Jumlah keseluruhan adalah 510 huruf yang terbagi ke dalam 3 data skenario data. Hasil tertinggi yang diperoleh mencapai 94,7% dengan menggunakan SVM kernel polynomial dan radial basis function. Sedangkan pada kata, pengenalan Hangul dilakukan pada kata Hangul tipe 1 dan 2 dengan 15 pola berbeda. Perbedaan pola ini didapat dengan merubah jenis font-nya. Akurasi tertinggi mencapai 82% dengan menggunakan SVM kernel polynomial saat menggunakan tipe data 1 dan skenario data 2. Tingkat keberhasilan pengenalan dipengaruhi oleh banyaknya data yang dilatih.

Kata kunci: Pengenalan Citra, Support Vector Machine, Hangul, kernel polynomial, kernel radial basis function

SUMMARY

Rahmatina Hidayati, Department of Electrical Engineering, Faculty of Engineering Universitas of Brawijaya, July 2018, *Hangul Recognition Using Support Vector Machine*, Academic Supervisor: Moechammad Sarosa dan Panca Mudjirahardjo.

The recognition of Hangul Image is more difficult compared with the recognition of Latin Image. It could be seen from how to arrange it. The Hangul is arranged from two dimensions, while the Latin is arranged from the left to the right. In the research, it is made a system to identify the Hangul image become the Latin text which is hoped to be used as a learning material how to read the Hangul itself. In general, the image recognition system is divided into three steps, the first is preprocessing that includes the tresholding, the segmentation which is done using the method of connected component-labeling, and thinning to decrease some information of a pattern and it uses Zhang Suen. The second, the system does taking the feature from every image. The method which is applied in looking at the feature is chain code. Finally, the recognition process utilizes Support Vector Machine (SVM) with some kernels (linear, polynomial, radial basis function, and laplacian). It is done the recognition to letter image and Hangul word. For the letter consists of 34 letters, where each letter has 15 different patterns. The amount of all is 510 which is divided into 3 data scenarios. The highest result which is achieved 94,7% by using SVM kernel polynomial and radial basic function.

While in the word, the recognition of Hangul is done to the type 2 Hangul word with 15 different patterns. The difference of these patterns are achieved by changing the type of font. The highest accurate is achieved by using the SVM kernel polynomial, is 82% when using data type 1 and data scenario 2. The level of recognition result is influenced by many trained data.

Keyword: Image recognition; Support Vector Machine; Hangul; kernel polynomial; kernel radial basis function

KATA PENGANTAR

Dengan memanjatkan puji syukur kehadirat Allah SWT atas limpahan rahmat dan hidayatNya penulis telah dapat menyelesaikan tesis dengan judul: "Pengenalan Hangul Menggunakan *Support Vector Machine"*. Diharapkan hasil penelitian ini nantinya dapat dijadikan pembelajaran bagaimana membaca Hangul dalam suatu citra. Sangat disadari bahwa dengan kekurangan dan keterbatasan yang dimiliki penulis, penelitian ini masih belum sempurna. Oleh karena itu, penulis mengharapkan saran yang membangun agar tulisan ini bermanfaat bagi yang membutuhkan.

Malang, Juli 2018

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Optical Character Recognition (OCR) merupakan sistem pengenalan karakter dengan *input* berupa citra. Dalam citra *input* tersebut, terdapat teks yang nantinya akan dikonversi menjadi teks yang bisa diedit (Seethalakshmi, et al, 2005). Kinerja sistem OCR ini sangat tergantung pada jenis teks dokumen yang diproses. Secara umum, teks dalam citra dikategorikan dalam 3 jenis, yakni tulisan tangan, *print,* dan yang diketik (Singh, et al, 2015).

Beberapa penelitian telah dilakukan dalam rangka membuat sistem OCR. Salah satu metode yang pernah digunakan adalah *Support Vector Machine* (SVM). Dan jenis karakter yang pernah diteliti dengan metode SVM ini antara lain angka India yang dikenal dengan Kannada *numeral* dan alfabet Latin *upper* dan *lower case* A-Z. Dalam penelitian ini, metode SVM akan diterapkan dengan data yang berbeda. Data tersebut adalah karakter bahasa Korea yang dikenal dengan Hangul.

Penelitian terkait dengan pengenalan Hangul sudah pernah dilakukan oleh Kyung-Won dan Jin H. Penulis menerapkan *Stochastic Relationship Modeling* untuk proses pengenalan tulisan tangan suku kata Hangul. *Input* penelitian berupa citra Hangul dengan *output* Hangul itu sendiri.

Dibandingkan dengan Latin, pengenalan Hangul cenderung lebih sulit. Hangul tersusun dari dua dimensi, sedangkan Latin tersusun dari kiri ke kanan (Kyung-Won & Jin H., 2003). Sejauh ini, penelitian mengenai pengenalan Hangul dilakukan dengan *output* berupa karakter Hangul. Dalam penelitian ini, akan dikembangkan pengenalan citra Hangul dengan *output* berupa karakter Latin. Konversi citra Hangul menjadi karakter Latin bisa digunakan sebagai pembelajaran bagaimana membaca Hangul itu sendiri.

Dalam sistem OCR, pada umumnya terbagi ke dalam 3 langkah, yakni *preprocessing,* ekstraksi fitur, dan pengenalan. *Preprocessing* meliputi 3 tahapan, pertama *tresholding* atau binerisasi, yakni merubah citra *grayscale* menjadi hitam-putih. Kemudian segmentasi yang akan memberikan label pada masing-masing huruf. Segmentasi dilakukan menggunakan *connected component-labeling*.

Tahap ketiga *thinning* atau penipisan yang berguna mengurangi sejumlah informasi sebuah pola menjadi garis tipis agar lebih efisien dalam proses pengambilan fitur(Lam, et al, 1992). Dalam hal ini akan diterapkan algoritma *thinning* paralel yang memiliki kinerja lebih cepat dibandingkan dengan algoritma lain. Algoritma tersebut dikenal dengan Zhang-Suen (Haseena, et al, 2017). Dalam prosesnya, algoritma Zhang-Suen melakukan pengecekan beberapa kondisi terhadap ketetanggan piksel. Jika memenuhi kondisi-kondisi algoritma, maka piksel tersebut akan dihapus. Namun jika sebaliknya, maka piksel akan tetap dipertahankan.

Setelah *preprocessing*, tahap selanjutnya adalah ekstraksi fitur yang memiliki peranan penting dalam pengenalan pola. Pada proses ini, akan didapat komponen dasar dari citra yang disebut dengan fitur-fitur (Nasien, et al, 2010). Dalam penelitian ini, ekstraksi fitur yang akan digunakan adalah *chain code* atau kode rantai yang menerapkan pencarian arah. Agar bisa dilakukan pengenalan, ekstraksi fitur masing-masing citra harus berjumlah sama. Untuk itu, perlu dilakukan normalisasi fitur.

Setelah mendapatkan fitur yang ternormalisasi, proses selanjutnya adalah pengenalan dengan menggunakan metode SVM. Ada dua tahapan dalam metode SVM, yakni pelatihan dan pengujian. Pelatihan dilakukan terhadap fitur-fitur citra untuk memperoleh model pemisah terbaik. Kemudian model pemisah tersebut diterapkan pada proses pengujian.

Pada prinsipnya, metode SVM hanya bisa diterapkan pada 2 kelas. Sedangkan dalam pengenalan Hangul, terdapat 34 huruf yang berarti ada 34 kelas. Untuk mengatasi permasalahan SVM agar bisa diterapkan pada lebih dari 2 kelas, diperlukan suatu fungsi yang dikenal dengan fungsi *kernel*. Dalam penelitian, *kernel* yang akan digunakan adalah *linear*, *polynomial*, *radial basis function*, dan *laplacian*.

1.2 Rumusan Masalah

Rumusan masalah yang akan diteliti berfokus pada:

1) Bagaimana menerapkan *connected component-labeling* untuk proses segmentasi?
2) Bagaimana menerapkan metode Zhang-Suen dalam proses penipisan?
3) Bagaimana menerapkan *chain code* untuk menemukan fitur-fitur dari masing-masing huruf dan kata Hangul dengan pencarian arah?
4) Bagaimana membangun sistem pengenalan huruf dan suku kata Hangul dengan menggunakan SVM *kernel linear, polynomial, radial basis function,* dan *laplacian*?

1.3 Tujuan Penelitian

Berdasarkan dari rumusan masalah di atas dan untuk mengukur keberhasilan penelitian, perlu ditetapkan tujuan penelitian sebagai berikut:

1) Menguji metode SVM (*kernel linear, polynomial, radial basis function*, dan *laplacian*) untuk sistem pengenalan Hangul.
2) Menerapkan metode *connected component labeling* untuk proses segmentasi.
3) Menerapkan metode *Zhang-Suen* untuk proses *thinning*.
4) Menerapkan metode *chain code* untuk mengekstraksi fitur dari masing-masing huruf dan kata Hangul.

1.4 Batasan Masalah

Agar pembahasan menjadi terfokus, dibuat batasan masalah sebagai berikut:

1) Data *input* berupa citra huruf dan kata Hangul yang diketik (*typewritten*).
2) Citra *input* berformat bitmap.
3) Ukuran citra yang digunakan adalah 75×70 piksel.
4) Data berupa 34 huruf di mana maing-masing terdiri dari 15 pola (total 510 huruf).
5) Data juga berupa kata dengan 2 tipe di mana setiap tipe terdiri dari 15 pola berbeda. Jumlah setiap tipe adalah tipe pertama 10 kata (total 150), tipe kedua 20 kata (total 300).
6) Sistem akan diimplementasikan menggunakan bahasa pemrograman C#.

Halaman ini sengaja dikosongkan

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Beberapa penelitian sudah dilakukan dalam rangka pengenalan huruf dengan menggunakan SVM. (Rajashekararadhya dan Ranjan, 2009) menerapkan metode SVM *kernel gaussian* pada data *Kannada* dengan fitur *Image Centroid* & *Zone* (ICZ). Akurasi yang diperoleh dengan menggunakan citra ukuran 50x50 adalah 97,25% untuk 400 data *testing*.

(Kumar, et al, 2012) mencoba membandingkan metode SVM dengan MLP *Neural Network* dalam sistem pengenalan huruf kapital A-Z. Dari perbandingan tersebut, metode SVM dengan *kernel linear* dan *polynomial* mendapatkan akurasi lebih tinggi dibandingkan MLP, yakni 94,8%.

(Tran, 2010) mengimplementasikan SVM *radial basis function* pada pengenalan karakter Perancis beraksen *upper case* (A-Z) dan *lower case* (a-z). Akurasi yang didapatkan dari kedua data tersebut adalah 95,6% dan 93,3%. Sedangkan (Nasien, dkk, 2010) menerapkan SVM fungsi *kernel radial basis function* pada pengenalan karakter latin *lower case*, *upper case*, serta kombinasi dari keduanya. Metode yang digunakan untuk mendapatkan fitur dari masing-masing huruf adalah *chain code*. Tingkat akurasi yang diperoleh *lower case* 86%, *upper case* 88,46%, dan kombinasi 73,44%.

Rangkuman dari penelitian penerapan metode SVM pada sistem pengenalan karakter terdapat dalam Tabel 2.1.

Tabel 2.1 Penelitian Terkait

No.	Judul	Penulis	Tahun	*Topic of Interest*	Kesimpulan
1.	*Support Vector Machine based Handwritten Numeral Recognition of Kannada Script*	S. V. Rajashekararadhya dan P. Vanaja Ranjan	2009	• Data yang digunakan berupa tulisan tangan Kannada numeral (angka India) dari 0-9 dengan jumlah 4000 dataset yang terdiri dari 2000 contoh dari 200 orang dan 2000 contoh dari 40 orang. • Ukuran data: 42x32, 50x50, dam 64x64 • Format data: bmp • Ekstraksi fitur: Image Centroid & Zone (ICZ) • Klasifikasi: (SVM) *kernel Gaussian*	• Akurasi rata-rata dari semua angka ukuran 50x50 untuk 2000 data training dan untuk data testing: - 400: 97, 25% - 800: 96,25% - 1200: 96,5% - 1600: 96,75% - 2000: 97,05%
2.	*Handwritten Character Recognition using Different Kernel based SVM Classifier and MLP Neural Network*	Parveen Kumar, Nitin Sharma, dan Arun Rana	2012	• Membandingkan SVM dan MLP • Data: tulisan tangan alphabet capital A-Z • Fungsi *kernel* linear dan polynomial digunakan dalam metode SVM	• Akurasi SVM 94, 8%, sedangkan MLP hanya 80,97%
3.	*Accented Handwritten Character Recognition Using SVM – Application to French*	De Cao Tran, Patrick Franco, Jean-Marc Orgier	2010	• Data: 1) kombinasi *database* UNIPEN dan IRONOF, terdiri dari tulisan tangan angka 0-9, *uuper case* (A-Z) dan *lower case* (a-z). 2) Tulisantangan karakter Perancis beraksen yang didapat dari 20 orang. Setiap angka ditulis 10 kali. • Artificial *database*: kombinasi 26 non-aksen karakter dan 13 karakter Perancis berjumlah 4000 contoh dari-masing-masing karakter. • *Local collection:* 200 contoh untuk setiap aksen yang juga digunakan untuk *training*. • Menggunakan 45 fitur dari 254 pilihan fitur yang diketahui dari 6iterature. • Metode: SVM dengan fungsi *kernel radial basis function.*	Akurasi • Angka 0-9: 98,7% • *Upper case* (A-Z): 95,6% • *Lower case* (a-z)*:* 93,3% • *4 kelas aksen*: 99,9% • Tes untuk 26 kelas *lower case* dan 4 kelas aksen karakter, didapatkan akurasi 93,5% untuk *artificial database*, dan 92, 7% untuk *local collection.*

No.	Judul	Penulis	Tahun	*Topic of Interest*	Kesimpulan
4.	*Support Vector Machine (SVM) For English Handwritten Character Recognation*	(Nasien, *et al*, 2010)	2010	• Data set: TD1 terdiri dari 189.411 huruf *lowercase*, TD2 berupa 217.812 *upper case*, dan TD3 berisi 407.233 kombinasi *lower case* dan *upper case*. • Data set dibagi menjadi 20% data *training* dan 80% *testing*. • Ekstraksi fitur: *chain code* • Metode: SVM dengan fungsi *kernel Radial Basis Function*.	• Akurasi yang diperoleh untuk masing-masing data set: TD1 86%, TD2 88,46%, dan TD3 73,44%.
5.	Utilization of Hierarchical Stochastic Character Recognition	(Kyung-Won, Kang & Jin H., Kim, 2004)	2004	• Data: KU-I *database* yang terdiri 1000 set dari 520 kata karakter Hangul yang paling sering digunakan. • Setiap set terdiri 520 karakter pola yang ditulis satu orang. • 200 set digunakan untuk *training*. Dan 100 set untuk *recognition*. • Dari 11.172 jenis karakter/kata modern Hangul, hanya 2.350 yang digunakan. • Metode: *Stochastic Relationship Modeling*.	• Jika target dibatasi 520, akurasi mencapai 90,3%. Dan 87,7% untuk target 2.350.

2.2 Dasar Teori

2.2.1 Pengenalan Citra Digital

Citra digital adalah citra yang dinyatakan dalam kumpulan data ditigal dan dapat diproses oleh komputer. Citra di dalam komputer disusun atas sejumlah piksel. Sebuah piksel dapat dibayangkan sebagai sebuah titik. Setiap piksel mempunyai koordinat, yang dinyatakan dengan bentuk (x,y) dengan y menyatakan baris dan x menyatakan kolom. Umumnya, koordinat pojok kiri atas dinyatakan dengan (0,0). Dengan demikian, jika suatu citra berukuran M baris dan N kolom atau biasa dinyatakan dengan M × N, koordinat piksel terbawah dan terkanan berada di koordinat (M-1, N-1).

Secara prinsip, citra dibagi menjadi tiga jenis, yaitu biner, citra berskala keabuan, dan citra warna. Citra biner atau dikenal dengan citra hitam-putih atau citra monokrom adalah citra yang nilai piksel-pikselnya berupa angka nol atau satu saja atau dua keaadaan seperti 0 dan

255. Citra seperti ini biasa dipakai untuk kepentingan segmentasi, yang memisahkan objek dari latar belakangnya.

Citra berskala keabuan *(grayscale)* adalah citra yang menggunakan gradasi warna abu-abu yang merupakan kombinasi antara hitam dan putih. Setiap warna di dalam citra berskala keabuan dinyatakan dengan sebuah nilai bulat antara 0 dan 255 (untuk yang aras keabuannya sama dengan 256) dan nilai tersebut sebagai intensitas.

Citra berwarna (*true color*) mempresentasikan keadaan visual objek-objek yang bisa kita lihat. Warna objek direkam. Citra berwarna atau dikenal sebagai RBG tersusun atas tiga komponen, yaitu komponen merah (R atau *Red*), komponen hijau (G atau *Green*), dan komponen biru (B atau *Blue*). Setiap piksel akan diwakili oleh tiga komponen tersebut (Kadir, A., 2013).

2.2.2. Hangul

Huruf Hangul merupakan satu kumpulan simbol fonetik atau huruf yang disebut Jaso yang dikombinasikan ke dalam bentuk karakter. Kumpulan ini terdiri dari 10 vokal dan 14 konsonan sebagai huruf dasar, dan beberapa dari mereka adalah huruf campuran yang ditunjukkan dalam Tabel 2.2.

Tabel 2.2 Huruf Hangul Jaso yang diklasifikasikan sesuai bentuknya

Huruf Dasar	Vokal	ㅏ ㅑ ㅓ ㅕ ㅣ ㅗ ㅛ ㅜ ㅠ ㅡ (10)
	Konsonan	ㄱ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ (14)
Huruf Campuran	Vokal	ㅐ ㅖ ㅔ ㅖ ㅘ ㅙ ㅚ ㅝ ㅞ ㅟ ㅢ (11)
	Konsonan	ㄲ ㄸ ㅃ ㅆ ㅉ ㄳ ㄵ ㄶ ㄺ ㄻ ㄼ ㄽ ㄾ ㄿ ㅀ ㅄ (16)

Sumber: Jin Ho (1995, p.760)

Hangul memiliki gabungan sistem karakter yang terdiri dari konsonan pertama (FC atau *first consonants*), vokal tengah (MV atau *middle vowels*), dan opsional konsonan terakhir (LC atau *last consonants*) untuk mewakili satu suku kata. Hal ini merupakan sistem karakter unik dalam Hangul yang diperlukan aturan dan metode berbeda dibandingkan bahasa berbasis huruf untuk pengenalan, analisis serta turunan karakter.

Tabel 2.3 menampilkan huruf Hangul yang terdiri dari 19 FC, 21 MV, dan 27 LC. Jika daftar karakter Hangul yang dikenali satu kata sebagai kombinasi dari [FC + MV] dan [FC + MV + LC], maka secara keseluruhan dimungkinkan 11.172 kata (Suyun Ju & Jungpil Shin, 2013).

Tabel 2.3 Huruf Hangul dan Huruf Latinnya

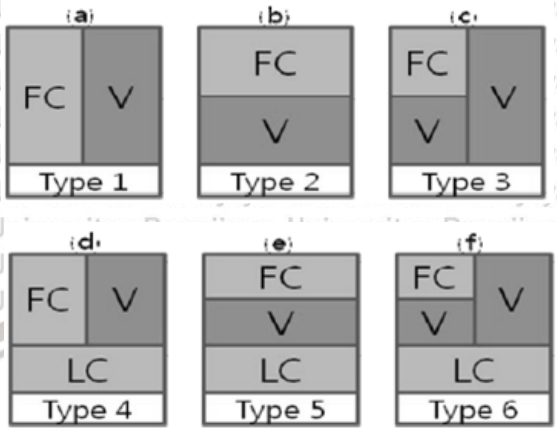
Huruf	*First consonants* (FC)		*Middle vowels (*MV)		*Last consonants* (LC)	
	Hangul	Latin	Hangul	Latin	Hangul	Latin
	ㄱ	g	ㅏ	a	ㄱ	g
	ㄴ	n	ㅐ	ae	ㄴ	n
	ㄷ	d	ㅑ	ya	ㄷ	d
	ㄹ	l	ㅒ	yae	ㄹ	l
	ㅁ	m	ㅓ	eo	ㅁ	m
	ㅂ	b	ㅔ	e	ㅂ	b
	ㅅ	s	ㅕ	yeo	ㅅ	s
	ㅇ	null/g	ㅖ	ye	ㅇ	null/g
	ㅈ	j	ㅣ	i	ㅈ	j
	ㅊ	ch	ㅗ	o	ㅊ	ch
	ㅋ	k	ㅛ	yo	ㅋ	k
	ㅌ	t	ㅜ	u	ㅌ	t
	ㅍ	p	ㅠ	yu	ㅍ	p
	ㅎ	h	ㅡ	eu	ㅎ	h

Jenis Huruf	*First consonants* (FC)		*Middle vowels* (MV)		*Last consonants* (LC)	
	Hangul	Latin	Hangul	Latin	Hangul	Latin
	ㄲ	gg	ㅘ	wa	ㄲ	gg
	ㄸ	dd	ㅙ	wae	ㄳ	gs
	ㅃ	bb	ㅚ	oe	ㄵ	nj
	ㅆ	ss	ㅝ	wo	ㄶ	nh
	ㅉ	jj	ㅞ	we	ㄺ	lg
			ㅟ	wi	ㄻ	lm
			ㅢ	ui	ㄼ	lb
					ㄽ	ls
					ㄾ	lt
					ㄿ	lp
					ㅀ	lh
					ㅄ	bs
					ㅆ	ss
Jumlah	19	19	21	21	27	27

Sumber: Suyun Ju dan Jung Shin (2013,p.2232) dengan penambahan huruf latin.

Keseluruhan tampilan Hangul *glyphs* yang terdiri dari tiga komponen (FC, MV, LC) ditempatkan dalam satu kotak. Menurut penempatan ketiga komponen tersebut dapat dibagi

menjadi kelompok huruf *vertical* dan *horizontal*. Lebih rincinya, Hangul *glyphs* terbagi menjadi 6 tipe bentuk yang ditunnjukkan pada Gambar 2.1.



Gambar 2.1 6 Tipe Bentuk Kata Hangul
Sumber: Suyun Ju dan Jungpil Shin (2013,p.2232)

Contoh kata masing-masing tipe bisa dilihat di Tabel 2.4.

Tabel 2.4 Contoh Kata dari Masing-masing Tipe

Tipe	**Contoh**				
1	가	비	사	뼈	네
	/ga/	/bi/	/sa/	/bbyeo/	/ne/
2	구	소	부	오	쓰
	/gu/	/so/	/bu/	/o/	/sseu/
3	과	뭐	왜	귀	궤
	/gwa/	/mwo/	/wae/	/gwi/	/gwel/
4	한	형	먹	읽	백
	/han/	/hyeong/	/meog/	/ilg/	/baeg/
5	돈	육	눈	꽃	꼭
	/don/	/yug/	/nun/	/ggoch/	/ggog/
6	관	원	활		
	/gwan/	/won/	/hwal/		

Dalam penelitiannya, (Suyun Ju & Jungpil Shin, 2013) meminimalkan data *input* dengan memperinci tipe bentuk Hangul seperti yang terlihat pada Gambar 2.2. Penelitian tersebut menggunakan sebuah metode yang membedakan konsonan tunggal dan konsonan ganda pada posisi konsonan pertama dan kosonan terakhir. Cara itu bisa mengurangi 18 data *input* yaitu 5 konsonan ganda pada konsonan pertama dan 13 konsonan ganda pada konsonan terakhir. Perbedaan data Hangul antara 6 tipe dasar dengan 18 tipe baru terdapat pada Tabel 2.5.

Gambar 2.2 18 Tipe Bentuk Kata Hangul
Sumber: Suyun Ju dan Jungpil Shin (2013,p.2233)

Tabel 2.5 6 Tipe Dasar dan 18 Tipe Baru Huruf Hangul

Jika ada 6 tipe		Jika lebih diperinci menjadi 18 tipe	
First consonants	ㄱ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ ㄲ ㄸ ㅃ ㅆ ㅉ	*Consonants*	ㄱ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ
Middle vowels	ㅏ ㅑ ㅓ ㅕ ㅣ ㅗ ㅛ ㅜ ㅠ ㅡ ㅐ ㅖ ㅔ ㅖ ㅘ ㅙ ㅚ ㅝ ㅞ ㅟ ㅢ	*Vowels*	ㅏ ㅑ ㅓ ㅕ ㅣ ㅗ ㅛ ㅜ ㅠ ㅡ ㅐ ㅖ ㅔ ㅖ ㅘ ㅙ ㅚ ㅝ ㅞ ㅟ ㅢ
Last consonants	ㄱ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ ㄲ ㄳ ㄵ ㄶ ㄺ ㄻ ㄼ ㄽ ㄾ ㄿ ㅀ ㅄ ㅆ		

Sumber: Suyun Ju dan Jungpil Shin (2013,p.2233)

2.3 *Preprocessing*

Preprocessing data menggambarkan semua pengolahan yang dilakukan pada data mentah yang dipersiapkan untuk proses selanjutnya. Oleh karena itu, *preprocessing* merupakan langkah awal mengubah data menjadi format yang akan lebih mudah dan efektif untuk diproses (Alginahi, Yasser, 2010).

Dalam penelitian ini, tahap *preprocessing* melibatkan proses binerisasi atau *tresholding*, segmentasi, dan penipisan atau *thinning*.

2.3.1 *Thresholding*

Thresholding merupakan proses memisahkan informasi (objek) dari *background*. *Thresholding* dapat dikategorikan menjadi 2 bagian, yakni *global thresholding* dan *local*

thresholding. Metode *global thresholding* memilih satu *threshold* atau nilai ambang untuk keseluruhan citra dokumen yang seringkali didasarkan pada estimasi *background level* dari intensitas histogram citra. Oleh karena itu, ini dianggap sebagai operasi pengolahan titik.

Di sisi lain, *local thresholding* menggunakan nilai berbeda untuk setiap piksel sesuai dengan informasi lokal area. *Local thresholding* digunakan pada citra dokumen yang memiliki penerangan *background* yang tidak sama atau *background* kompleks, seperti adanya *watermak* (Alginahi, Yasser, 2010).

Proses *thresholding* akan menghasilkan citra biner, yaitu citra yang memiliki dua tingkat keabuaan yaitu hitam dan putih. Secara umum proses pengambangan citra *grayscale* untuk menghasilkan citra biner adalah sebagai berikut (Putra, 2010).

$$g(x,y) = \begin{Bmatrix} 1\ if\ f(x,y) \geq T \\ 0\ if\ f(x,y) < T \end{Bmatrix} \quad (2\text{-}1)$$

Dengan g(x,y) adalah citra biner dari citra *grayscale* f(x,y) dan T menyatakan nilai ambang. Nilai T memegang peranan yang sangat penting dalam proses pengambangan. Kualitas hasil citra biner sangat tergantung pada nilai T yang digunakan (Putra, 2010).

2.3.2 Segmentasi *Connected Component-Labeling*

Segmentasi citra bertujuan untuk mendapatkan objek-objek yang terkandung di dalam citra atau membagi citra ke dalam beberapa daerah dengan setiap objek daerah memiliki kemiripan atribut. Pada citra yang mengandung hanya satu objek, objek dibedakan dari latar belakangnya (Kadir & Susanto, 2013).

Ekstraksi *connected component* antar piksel dalam citra biner merupakan langkah mendasar dalam segmentasi objek dan *region* citra. atau blob (gumpalan). Dalam citra biner, analisis objek biasanya diekstraksi dengan menggunakan operasi *connected component labeling* yang terdiri dari pemberian label unik ke masing-masiing *region* yang terhubung secara maksimal dari piksel latar depan *(foreground)* (Stefano, L. D. & Bulgarelli, A.).

2.3.2.1 Penandaan Komponen Terhubung (*Connected Componen- Labeling)*

Penandaan komponen terhubung memeriksa suatu citra dan mengelompokkan setiap piksel ke dalam suatu komponen terhubung menurut aturan keterhubungan (4,8, atau m-*connectivity*). Setiap komponen terhubung yang saling tidak terhubung *(disjoin)* pada suatu

citra akan diberi tanda (label) berbeda. Memisahkan dan memberikan tanda pada setiap komponen terhubung maupun tidak terhubung pada sutau citra memegang peranan sentral pada beberapa aplikasi analisis citra secara otomatis.

Penandaan komponen terhubung dilakukan dengan memeriksa suatu citra, piksel per piksel (dari kiri ke kanan dan atas ke bawah) untuk mengidentifikasi area piksel terhubung yaitu area dari piksel berbatasan yang memiliki nilai intensitas sama atau nilai intensitas berada dalam suatu himpunan V (pada citra biner V = {1}, pada citra keabuan himpunan V disesuaikan dengan kebutuhan). Penandaan komponen terhubung dapat dilakukan pada citra biner maupun citra keabuan.

Penandaan komponen terhubung untuk aturan 4-*connectivity* berbeda dengan 8-*connectivity*. Berikut dijelaskan tahapan dalam melakukan penandaan komponen terhubung pada masing-masing jenis keterhubungan berikut (Putra, 2010).

a. 4-*connected*

- Periksa (*scan*) citra dengan bergerak sepanjang baris sampai menemukan piksel p (nilai p berada dalam himpunan V). Bila p sudah ditemukan maka periksa nilai piksel tetangga dari p, yaitu piksel di atas dan di kiri dari p, kemudian lakukan pemeriksaan berikut.
- Bila kedua piksel tetangga bernilai 0 maka berilh tanda (label) baru pada p.
- Jika hanya satu saja dari piksel tetangga tersebut bernilai 1 maka berilah tanda dari piksel tetangga tersebut pada p.
- Bila kedua piksel tetangga bernilai 1 dan memiliki tanda sama maka berilah tanda dari piksel tetangga tersebut pada p.
- Bila kedua piksel tetangga bernilai 1 dan memiliki tanda berbeda maka berilah tanda dari salah satu piksel tetangga tersebut pada p dan buat catatan bahwa kedua tanda yang berbeda tersebut adalah ekuivalen.

Pada akhir proses, semua piksel bernilai 1 (untuk citra biner) telah mendapat tanda tetapi beberapa tanda tersebut mungkin masih ada yang ekuivalen. Oleh karena itu proses berikutnya yang harus dilakukan adalah mengurutkan pasangan-pasangan tanda yang ekuivalen ke dalam kelas-kelas ekuivalen kemudian memberi tanda yang berbeda pada setiap kelas ekuivalen.

b. 8-*connected*

Penandaan komponen terhubung 8-*connected* hamper sama dengan 4-*connected* hanya saja piksel tetangga yang diperiksa selain piksel atas dan kiri, juga kedua piksel diagonal atas dari p, sehingga ada 4 piksel tetangga p yang diperiksa yaitu, atas, kiri, diagonal atas kiri, dan diagonal atas kanan dengan ketentuan berikut.

- Bila keempat piksel tetanggaan bernilai 0 maka berilah tanda baru pada p.
- Bila hanya salah satu piksel tetangga bernilai 1 maka berilah tanda dari piksel tetangga tersebut pada p.
- Bila dua atau lebih piksel tetangga bernilai 1 maka berilah salah satu tanda dari piksel tetangga tersebut pada p, kemudian buat catatan bahwa semua tanda dari piksel tetangga bernilai 1 tersebut adalah ekuivalen.

Proses berikutnya adalah membuat kelas-kelas ekuivalen seperti pada 4-*connected* dan member setiap kelas ekuivalen tanda yang berbeda. Langkah terakhir dari proses penandaan baik untuk 4 atau 8-*connected* adalah melakukan pemeriksaan (*scanning*) kembali pada citra dang anti setiap tanda dengan tanda dari kelas ekuivalen.

2.3.3 *Thinning* Zhang-Suen

Thinning merupakan suatu operasi morfologi, terkadang seperti erosi atau *opening*. *Thinning* mengubah bentuk asli citra biner menjadi citra yang menampilkan batas-batas objek/*foreground* hanya setebal satu piksel. Sepintas, *thinning* mempunyai kemiripan dengan deteksi tepi dalam hal *output* dari citra yang dihasilkan. Kedua proses tersebut sama-sama menampilkan batas objek pada citra. Namun, tetap saja ada perbedaan antara *thinning* dan deteksi tepi dari sisi cara kerjanya sebagai berikut (Putra, 2010).

- Deteksi tepi: merubah *gray* level atau intensitas citra menjadi citra yang menampilkan batas-batas/*boundaries* obyek berdasarkan kekontrasan warna antar piksel.
- *Thinning:* mereduksi piksel pada obyek biner menjadi piksel yang bernilai sama dengan piksel pada *background*. Keluaran berupa citra biner dengan informasi berupa batas-batas obyek berdasarkan piksel dengan ketebalan satu piksel.

Thinning berguna mengurangi sejumlah informasi sebuah pola menjadi garis tipis agar lebih mudah dianalisa. *Thinning* juga dikenal dengan proses *skeletonization* yang mana keluarannya disebut *skeleton* atau kerangka (Lam, *et al*, 1992).

Ada dua pendekatan pada proses *thininng*, yakni non iteratif dan iteratif. Teknik non iteratif menghasilkan sebuah *skeleton* secara langsung tanpa memeriksa semua piksel secara individu. Beberapa metode yang menggunakan pendekatan teknik ini adalah *neural network*, Vonoral diagram, dan *wavelet transform* (Abu-Ain, *et al*, 2013).

Thining dengan teknik iteratif menghapus secara berturut-turut lapisan-lapisan piksel pada batas pola sampai hanya menyisakan sebuah kerangka. Penghapusan atau penyimpanan piksel tergantung dari konfigurasi piksel pada ketetanggaan lokal. Berdasarkan cara pemeriksaan piksel, algoritma ini diklasifikasi menjadi sekuensial dan paralel (Lam, *et al*, 1992).

Algoritma sekuensial dan paralel memiliki kesamaan dalam menentukan piksel yang diinginkan dan tidak diinginkan, Namun, berbeda pada waktu penghapusan. Dalam sekuensial, penghapusan piksel yang tidak diinginkan dimulai pada saat proses identifikasi yang diinginkan. Sedangkan dalam paralel, piksel dihapus setelah mengindentifikasi keseluruhan piksel yang tidak diinginkan (Abu-Ain, *et al*, 2013).

Hasil review beberapa algoritma *thinning*, ditetapkan bahwa Zhang-Suen memiliki kinerja lebih cepat dibandingkan dengan algoritma *thinning* paralel yang lain (Haseena, M. H. F & Clara, A. R., 2017). Algoritma ini bekerja menggunakan matriks 3×3 dengan 8 ketetanggaan dari piksel (i,j), di mana piksel ketetanggaannya diasumsikan seperti pada Gambar 2.3.

P_9 (i-1,j-1)	**P_2 (i-1,j)**	**P_2 (i-1,j+1)**
P_8 (i,j-1)	**P_1 (i,j)**	**P_4 (i,j+1)**
P_7 (i+1,j-1)	**P_6 (i+1,j)**	**P_5 (i+1,j+1)**

Gambar 2.3 Desain 9 piksel dalam 3×3
Sumber: Zhang dan Suen (1984,p.236)

Untuk menjaga konektivitas dari *skeleton*, algoritma Zhan-Suen membagi setiap iterasi menjadi 2 sub-iterasi. Pada iterasi pertama, piksel P akan dihapus dari pola jika memenuhi kondisi berikut (Zhang, T.Y. & Suen, C. Y., 1984):

(a) $2 \leq B(P1) \leq 6$ (2-2)

(b) $A(P1) = 1$ (2-3)

(c) $P2 \times P4 \times P6 = 0$ (2-4)

(d) $P4 \times P6 \times P8 = 0$ (2-5)

Di mana $B(P_1)$ adalah jumlah *foreground* dari ketetanggaan P_1. $A(P_1)$ merupakan jumlah pola 01 antara P_2, P_3, P_4, . . . P_8, P_9 yang merupakan ketetanggaan dari piksel P_1 atau jumlah transisi dari *background* ke *foreground*. Terlihat pada Gambar di atas, terdapat dua transisi sehingga nilai $A(P_1) = 2$. Berdasarkan Kondisi ini, piksel P tidak memenuhi syarat untuk dihapus.

Pada iterasi kedua, piksel P akan dihapus jika memenuhi kondisi:

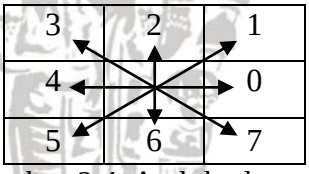
(a) dan (b) seperti iterasi pertama

(c) $P2 \times P4 \times P8 = 0$ (2-6)

(d) $P2 \times P6 \times P8 = 0$ (2-7)

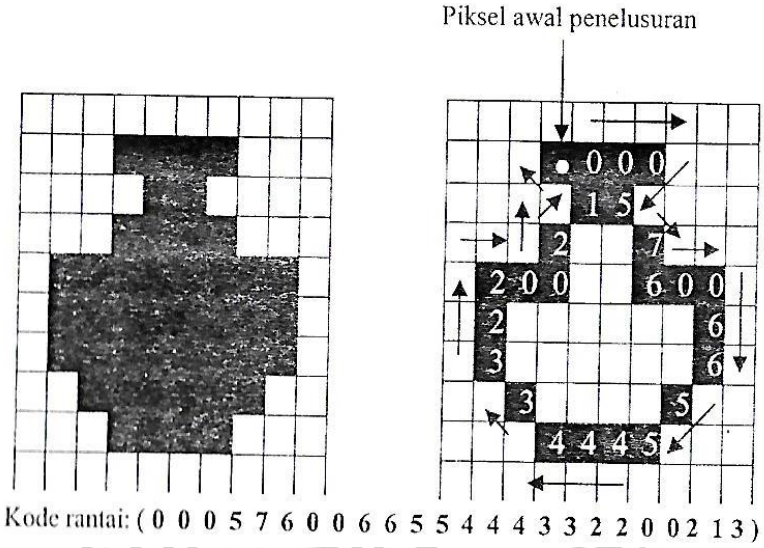
2.4 Ekstraksi Fitur *Chain Code*

Chain code atau kode rantai adalah suatu kode yang menunjukkan arah pergerakan dari perbatasan luar yang saling menyambung hingga membentuk rantai. Hal ini dapat dilakukan dengan menelusuri sekali lagi piksel-piksel perbatasan dari satu titik hingga kembali ke titik tersebut. Penomoran arah biasanya menggunakan aturan sebagai berikut (Sutoyo, dkk, 2009).



Gambar 2.4 Arah kode rantai
Sumber: Sutoyo, dkk (2009,p.142)

Cara membuat kode rantai adalah dengan memeriksa ke mana arah suatu piksel tersambung ke piksel lainnya yang berstatus 8-tetangga dan masing-masing arah mempunyai nomor yang berbeda. Tanda titik menunjukkan piksel awal, yaitu tempat di mana akan ditentukan arah dari langkah berikutnya. Gambar menunjukkan objek dan kode rantainya (Sutoyo, dkk, 2009).



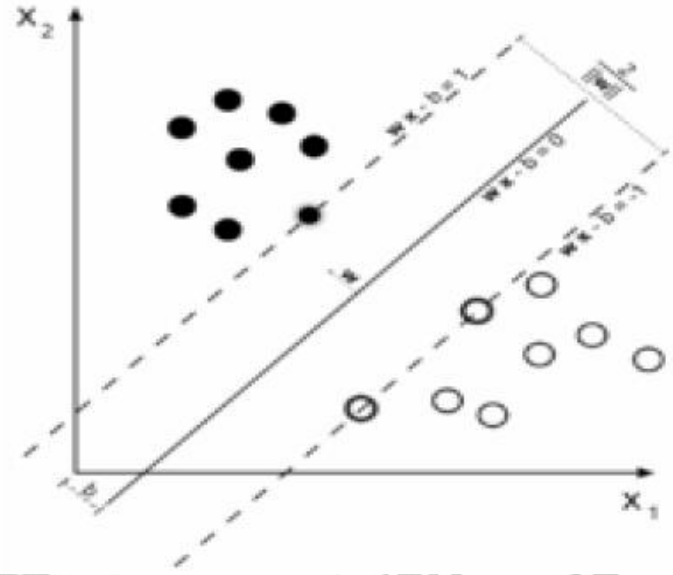
Gambar 2.5 Bentuk Objek dan Kode Rantainya
Sumber: Sutoyo, dkk (2009,p.143)

2.5 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) serta varian dan ekstensinya sering disebut metode berbasis *kernel* (atau hanya metode *kernel*) yang telah dipelajari secara ekstensif dan diterapkan pada berbagai klasifikasi pola dan pendekatan fungsi. Klasifikasi pola adalah mengklasifikasikan beberapa objek menjadi salah satu kategori yang disebut kelas (Abe, S., 2010).

Berdasarkan karakteristiknya, metode SVM terbagi menjadi 2, yaitu *SVM Linear* dan SVM *Non-Linear*. *SVM linear* adalah metode *SVM* yang memisahkan kedua kelas oleh *hyperplane* yaitu dengan *soft margin*. Sedangkan *SVM Non-linear* merupakan metode SVM yang menerapkan fungsi *kernel trick* pada ruang berdimensi tinggi (Prasetyo, 2014).

Pada metode *SVM linear*, masukan data dianggap sebagai dua kumpulan vektor dalam n dimensi ruang. *SVM* akan membangun sebuah pemisah *hyperplane* dalam ruang tersebut, salah satunya memaksimalkan "*margin*" antara dua kumpulan data. Untuk menghitung *margin*, buat dua paralel *hyperplane*, masing-masing satu di setiap sisi pemisah yang didorong melawan dua kumpulan data. Secara intuitif , pemisahan yang baik dicapai oleh *hyperplane* yang memiliki jarak terbesar untuk titik-titik data yang berdekatan dari kedua kelas. Gambar 2.6 menampilkan pemisahan dua kelas secara *linear* (Aroral, *et al*, 2010).



Gambar 2.6 SVM dengan Pemisah *Linear*
Sumber: Aroral, *et al* (2010,p.43)

Pemisahan yang baik dicapai *hyperplane* yang memiliki jarak terbesar untuk titik ketetanggaan dari dua kelas. Kumpulan data latih { $(x_1, y_1), (x_2, y_2), \ldots (x_i, y_i)$}, y bernilai 1 dan -1, menunjukkan kelas data yang dimilki. Berikut persamaan bidang pemisah atau *hyperplane* bisa dilihat pada Persamaan (2-8)(Aroral, *et al*, 2010).:

$$x_i \,.\, w_i + b \geq 1 \text{ untuk } y_i = +1 \quad (2\text{-}8)$$

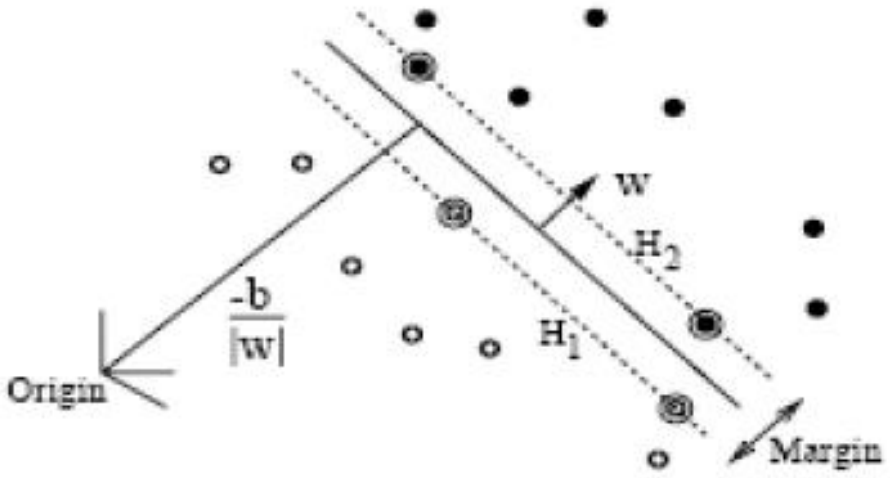
$$x_i \,.\, w_i + b \geq -1 \text{ untuk } y_i = -1 \quad (2\text{-}9)$$

Di mana x_i adalah data ke-i, w merupakan *support vector* yang tegak lurus terhadap *hyperplane* atau bidang normal antara bidang pemisah terhadap pusat koordinat. Parameter b (*bias*) adalah bidang relatif terhadap pusat koordinat yang menentukan *offset hyperplane* dari asal ke *vector* normal. Untuk menghitung w dan b dapat menggunakan persamaan (2-10) dan (2-11).

$$w = \sum_{i=1}^{n} = a_i \gamma_i x_i \quad (2\text{-}10)$$

$$b = -\frac{1}{2}(w.x\,(-) + w.x\,(+)) \quad (2\text{-}11)$$

-b/||w|| adalah jarak tegak lurus dari *hyperplane* ke asal yang ditunjukkan pada Gambar 2.7. Pilih w dan b untuk memaksimalkan *margin* atau jarak paralel *hyperplane* dipisah sejauh mungkin ketika memisahkan data.



Gambar 2.7 Hyperplane pemisahan *linear*
Sumber: Aroral, *et al* (2010,p.44)

SVM menemukan pemisah *hyperplane* optimal dengan menyelesaikan optimasi *quadratic programming*. Persamaan (2-12) merupakan model *quadratic* yang memakai klasifikasi *non-linear*. $K(x_i,x_j)$ adalah simbol persamaan *kernel trick*.

$$\arg\min \frac{1}{z}\|w\|^2 + C.\sum_{i=1}^{m} = a_i\, a_j y_i y_j\, K\left(x_i x_j\right) - \sum_{i=1}^{m} = a_i \qquad (2\text{-}12)$$

Dengan batasan:

$\alpha_i \geq 0$ ($i = 1, 2, \ldots, l$) $\Sigma\, \alpha_i\, y_i = 0$

Dari hasil perhitungan di atas akan didapatkan α_i yang kebanyakan bernilai positif. Data yang berhubungan dengan α_i positif inilah yang disebut sebagai *support vector* (Nugroho, dkk, 2003). Untuk memperoleh hasil klasifikasi yang optimal data uji x menggunakan Persamaan (2-13) dan Persamaan (2-14).

$$f(x)=(w.x+b) \text{atau } f(x) = C\ \sum_{i=1}^{m} = a_i y_i x_i\ (x + b) \qquad (2\text{-}13)$$

Fungsi klasifikasi = *sign (f(x))* (2-14)

Jika bernilai positif maka *sign (f(x))* sama dengan *1*, sedangkan bila negatif maka *sign (f(x))* sama dengan-*1*.

Dengan menggunakan *kernel trick*, SVM akan memvisualkan data ke ruang dimensi tinggi dan menemukan pemisahan terbaik (Lin, 2011).

2.5.1 Metode Sekuensial SVM

Tahap pertama dalam metode sekuensial adalah menginisialisasi $a_i = 0$ yang bertujuan untuk mencari *support vector*. Setelah inisialisasi, langkah selanjutnya adalah mencari nilai *matriks Hessian* menggunakan Persamaan (2-15).

$$D_{ij} = y_i y_j \left(K\left(x_i x_j\right) + \lambda^2\right) \quad (2\text{-}15)$$

Pada Persamaan (2-12), x_i adalah data ke – i, x_j adalah data ke–j, y_i adalah kelas data ke–i, dan $y_j$adalah kelas data ke-j. $K\left(x_i x_j\right)$adalah fungsi *kernel* yang digunakan dan λ merupakan konstanta *lambda*. Setelah memperoleh nilai *Hessian*, tahapan berikutnya adalah mendapatkan kondisi iterasi maksimum dengan syarat $[\delta a] < \varepsilon$ (*epsilon*), melalui Persamaan (2-16) sampai (2-18).

$$E_i = \Sigma_j^i = a_i D_{i\,j} \quad (2\text{-}16)$$

$$\delta a = min\,\{maks[\,\gamma(1 - E_i\,),\, -\alpha_i\,],\, C - \alpha_i\,\} \quad (2\text{-}17)$$

$$\alpha = \alpha_i + \delta a \quad (2\text{-}18)$$

Di mana α_i adalah *alpha* ke–i, D_{ij} adalah *matriks Hessian*, dan E_i adalah *error rate*. Untuk nilai konstanta γ (*gamma*) dan C (*complexity*), sedangkan δa merupakan delta alfa ke–i.

Selanjutnya adalah memperoleh nilai *support vector* atau *weight (w)* menggunakan Persamaan (2-19).

$$\vec{w} = \sum_{i=1}^{l} a_i y_i \vec{x}_i \quad (2\text{-}19)$$

Pada Persamaan (2-16), a_i merupakan fungsi *lagrange*, $y_i$adalah konstanta, dan $\vec{x}_i$ adalah data *support vector*. Setelah memperoleh nilai *support vector* langkah selanjutnya adalah mencari nilai *bias* (b) atau posisi bidang relatif terhadap koordinat menggunakan Persamaan (2-20).

$$b = -\frac{1}{2}(< \vec{w} \,.\, \vec{x}_{-1} > + < \vec{w} \,.\, \vec{x}_{+1} >) \quad (2\text{-}20)$$

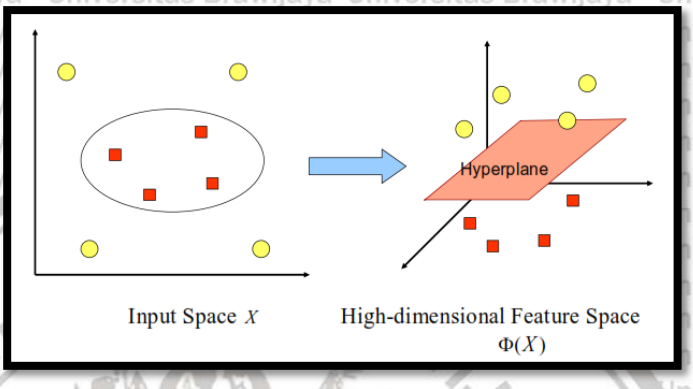
Di mana b adalah nilai *bias*, $\vec{w}$ adalah *weight* atau *support vector*, $\vec{x}_{-1}$ adalah data *support vector* dari kelas negatif, dan $\vec{x}_{+1}$ adalah data *support vector* dari kelas positif. Tahap terakhir adalah mendapatkan hasil klasifikasi dengan Persamaan (2-21).

$$f(x) = \sum_{i=1}^{m} a_i \mathrm{y_i} x_i \,(x + b) \quad (2\text{-}21)$$

Pada Persamaan (2-21), a_i adalah fungsi *lagrange* baru, b adalah nilai *bias*, dan y_i adalah konstanta.

2.5.2 *Kernel Trick*

Support Vector Machine (SVM) dikembangkan untuk dapat menyelesaikan masalah *non-linear* dengan memasukkan fungsi *kernel*. Dalam SVM *non-linear,* pertama-tama *(x)* dipetakan oleh fungsi $\phi(x)$ ke ruang vektor yang berdimensi lebih tinggi. Pada ruang vektor yang baru ini, *hyperplane* yang memisahkan kedua kelas tersebut dapat dikonstruksikan.



Gambar 2.8 Hyperplane Non-Linear
Sumber : Nugroho, dkk (2003)

Gambar 2.8 menampilkan fungsi Φ memetakan data ke *ruang vector* yang berdimensi lebih tinggi, sehingga kedua kelas dapat dipisahkan secara *linear* oleh sebuah *hyperplane*. Notasi matematika dari *mapping* ini seperti pada Persamaan (2-22).

$$\Phi : R^d \rightarrow R^d \quad (2\text{-}22)$$

Karena transformasi Φ umumnya tidak diketahui, maka bisa digunakan beberapa macam fungsi *kernel* sebagai berikut (Prasetyo, 2014) :

1. *Kernel Linear* ditunjukkan pada Persamaan (2-23).

$$K(x, y) = x.\ y \quad (2\text{-}23)$$

2. *Kernel Polynomial* ditunjukkan pada Persamaan (2-24).

$$K(x, y) = (x.\ y + c)^d \quad (2\text{-}24)$$

3. *Kernel Radial Basis Function* ditunjukkan pada Persamaan (2-25).

$$K(x, y) = exp\ (-\ ||x - y||^2 / 2\sigma^2) \quad (2\text{-}25)$$

4. *Kernel Laplacian* ditunjukkan pada Persamaan (2-26) (Fadel, *et al*, 2016).

$$K(x, y) = \exp\ (-||x - y|| / 2\sigma) \quad (2\text{-}26)$$

Di mana *x* dan *y* adalah pasangan dua data dari semua bagian data latih. Parameter σ *,c, d* > *0,* merupakan konstanta $||x\text{-}y||^2$ merupakan kuadrat jarak antara vektor *x* dan *y*.

Pemilihan fungsi *kernel* yang tepat merupakan hal yang sangat penting karena fungsi *kernel* ini akan menentukan fitur baru (dimensi tinggi) dimana fungsi klasifikasi (*hyperplane*) akan dicari. Sepanjang fungsi *kernel*-nya logis, SVM akan bekerja secara benar meskipun tidak diketahui seperti apa pemetaan yang dilakukan (Prasetyo, 2014).

2.5.3 *Multiclass Support Vector Machine* (M-SVM)

SVM hanya dapat melakukan klasifikasi biner (dua kelas) (Prasetyo, 2014). Sementara permasalahan di dunia nyata umumnya melibatkan banyak kelas dalam proses klasifikasi seperti pengenalan karakter, atau diagnosis pasien, di mana data masukan terbagi menjadi lebih dari dua kelas. Adapun beberapa pendekatan multikelas SVM seperti *one-against-all, one-against-one, error output code* (Tan, dkk,2005) (dalam Prasetyo, 2014).

2.5.3.1 Metode SVM *One Against One*

Metode SVM *one against one* adalah salah satu metode untuk mengimplementasi SVM untuk *multiclassproblem***.**Model pengenalan biner yang dibangun menggunakan metode ini dapat dihitung dengan mengikuti Persamaan (2-27) sampai Persamaan (2-29)(Nugroho, dkk, 2003).

$$\min \frac{1}{2}(w^{ij})^T w^{ij} + C \sum_t \xi_t^{ij} \quad (2\text{-}27)$$

Dengan meminimumkan dimensi Vapnik-Chervonenkis dengan persamaan $\min \frac{1}{2}(w^{ij})^T$, dimana w adalah bobot *support vector*, i dan j adalah jenis kelas. Sedangkan $C \sum_t \xi_t^{ij}$ merupakan persamaan untuk meminimalkan*error* pada pelatihan.C adalah konstanta yang dikalikan dengan variabel ξ_t^{ij} yaitu nilai *soft margin hyperplane*. Persamaan (2-28), bertujuan untuk mendapatkan bidang pemisah terbaik untuk SVM *non-linear*. Sehingga untuk mendapatkan kelas pertama menggunakan Persamaan (2-29) dan Persamaan (2-31) merupakan persamaan untuk mendapatkan kelas yang kedua, dengan kondisi ($\xi_t^{ij} \geq 0 : \xi_t^{ij} = 0$, jika klasifikasi benar).

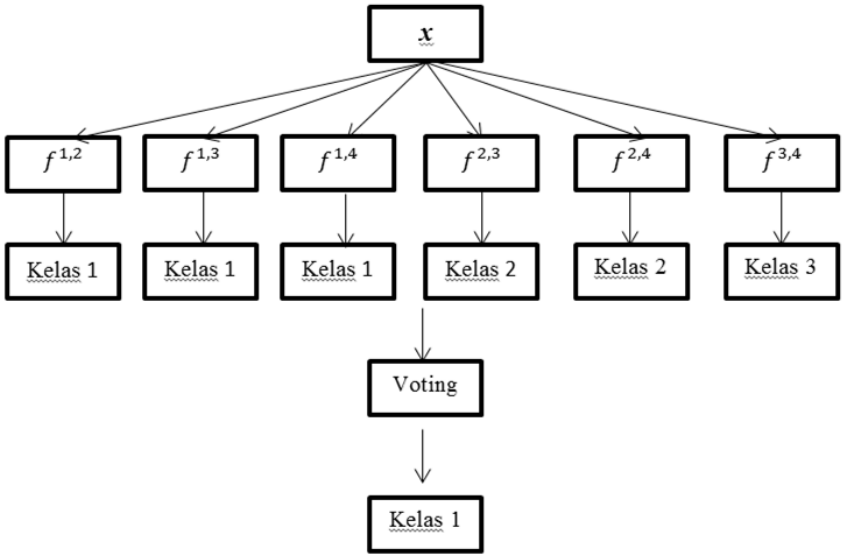
$$\left(w^{ij}\right)^T \emptyset\,(x_t) + b_{ij} > 1 - \xi_t^{ij} \rightarrow y_t = i, \quad (2\text{-}28)$$

$$\left(w^{ij}\right)^T \emptyset\,(x_t) + b_{ij} < -1 + \xi_t^{ij} \rightarrow y_t = j, \quad (2\text{-}29)$$

Dengan *k* adalah banyaknya kelas. Pada tahap pelatihan, setiap model pengenalan dilatih menggunakan data latih dari dua kelas. Sedangkan pada tahap pengujian terdapat beberapa cara untuk melakukan pengujian setelah semua *k(k −1)/2*model klasifikasi telah selesai dibangun. Contoh penggunaan metode SVM *One Against One* dengan pengenalan 4 kelas dapat ditunjukkan pada Tabel 2.5dan Gambar 2.9.

Tabel 2.6 Contoh Metode *One Against One*

$y_I = 1$	$y_I = -1$	Hipotesis
Kelas 1	Kelas 2	$f^{1,2}(x)=(w^{1,2})x+b^{1,2}$
Kelas 1	Kelas 3	$f^{1,3}(x)=(w^{1,3})x+b^{1,3}$
Kelas 1	Kelas 4	$f^{1,4}(x)=(w^{1,4})x+b^{1,4}$
Kelas 2	Kelas 3	$f^{2,3}(x)=(w^{2,3})x+b^{2,3}$
Kelas 2	Kelas 4	$f^{2,4}(x)=(w^{2,4})x+b^{2,4}$
Kelas 3	Kelas 4	$f^{3,4}(x)=(w^{3,4})x+b^{3,4}$



Gambar 2.9 Metode Pengenalan *SVM One Against One*

Sumber : Nugroho, dkk (2003)

Dari Gambar 2.9 jika data *xi*dimasukkan ke dalam fungsi yang didapatkan dari tahap pelatihan pada Persamaan (2-30).

$$f(x) = (w^{ij})^T \phi(x) + b \quad (2\text{-}30)$$

Dan hasil yang didapatkan *x* adalah kelas termasuk kelas*i*, maka kelas *i* mendapatkan *vote* atau suara. Dan selanjutnya data x_i diujikan ke semua model klasifikasi yang didapatkan dari tahap pelatihan. Pada akhirnya kelas dari data *x*ditentukan dari jumlah perolehan suara terbanyak.Apabila terdapat dua buah kelas yang memiliki jumlah suara yang sama, maka kelas dengan indeks yang lebih kecil dinyatakan sebagai kelas dari data yang diujikan.

2.5.3.2 Metode SVM *One Against All*

Metode ini akan membangun sejumlah *k* SVM biner, di mana *k* adalah banyaknya kelas (Hsu,dkk, 2002) (dalam Prasetyo, 2014). SVM ke-*i* dilatih dengan seluruh *sample* pada kelas ke-*i* dengan label kelas positif dan seluruh *sample* lainnya dengan label kelas negatif. Jika diberikan *l* data pelatihan $(x_1,y_1),\ldots,(x_l,y_l)$, dengan $x_i \in \mathrm{R}^n, \mathrm{i}=1,...,\mathrm{l}$ adalah kelas dari x_i, maka SVM ke-*i* akan menyelesaikan permasalahan berikut.

$$\left(w^i\right)^T \emptyset\left(x_j\right) + b^i < -1 + \xi_j^i \rightarrow y_j \neq i \quad (2\text{-}31)$$

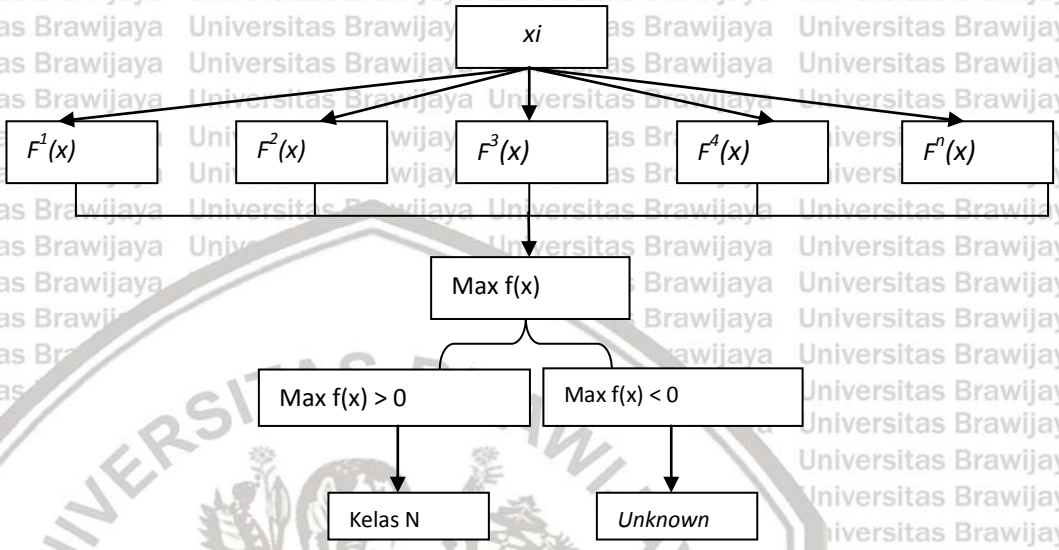
Pada Persamaan (2-31), *w* adalah bobot *support vector*, *i* dan *j* adalah kelas, *b* adalah nilai *bias*, dan $\xi_{\mathrm{j}}^{\mathrm{i}}$ adalah *soft margin*. Setelah menggunakan Persamaan (2-31), maka terdapat *k* fungsi dengan Persamaan (2-32).

$$f^1(x) = (w^1)x + b^1, \ldots, f^k(x) = (w^k)x + b^k \quad (2\text{-}32)$$

Tabel 2.7 Contoh Metode *One Against All*

$y_i = 1$	$y_i = -1$	Hipotesis
Kelas 1	Bukan kelas 1	$f^1(x) = (w^1)x + b^1$
Kelas 2	Bukan kelas 2	$f^2(x) = (w^2)x + b^2$
Kelas 3	Bukan kelas 3	$f^3(x) = (w^3)x + b^3$
Kelas 4	Bukan kelas 4	$f^4(x) = (w^4)x + b^4$
:	:	:
Kelas 34	Bukan kelas 34	$f^{34}(x) = (x^{34})x + b^{34}$

Tabel 2.7 menampilkan hipotesis dari metode *one against all* dengan cara kerja pemilihan kelas tersebut benar atau bukan. Seperti yang terdapat di kolom 1 pada Tabel 2.5, jika $y_i = 1$ maka dianggap sebagai kelas 1, namun jika bukan akan langsung dianggap bukan kelas tersebut.



Gambar 2.10 Metode Pengenalan *SVM One Against All*

Sumber : Nugroho, dkk (2003)

Pada Gambar 2.10 menjelaskan terdapat n fungsi keputusan, dari n fungsi keputusan diambil fungsi keputusan yang maksimal.

Halaman ini sengaja dikosongkan

BAB III
KERANGKA KONSEP PENELITIAN

3.1 Deskripsi Permasalahan

Pengenalan karakter adalah sebuah proses identifikasi karakter yang dilakukan dengan dua cara yaitu secara *online* dan *offline*. Lebih lanjut, salah satu pengenalan secara *offline* dikenal dengan *Optical Character Recognition* (OCR). Dalam hal ini OCR bisa dibagi ke dalam pengenalan karakter berupa *handwritten* dan *typewritten* (Dalbir & Singh, 2015).

Optical Character Recognition (OCR) merupakan sistem pengenalan karakter dengan *input* berupa citra. Dalam citra *input* tersebut, terdapat teks yang nantinya akan dikonversi menjadi teks yang bisa diedit (Seethalakshmi, et al, 2005).

Beberapa pengenalan karakter sudah dilakukan dengan menggunakan data berbeda. Umumnya huruf latin baik *upper case* (A-Z) ataupun *lower case* (a-z). Karakter lainnya berupa huruf yang memiliki keunikan sendiri, seperti huruf Hijayah, Katana, Hiragana, India, dan Hangul.

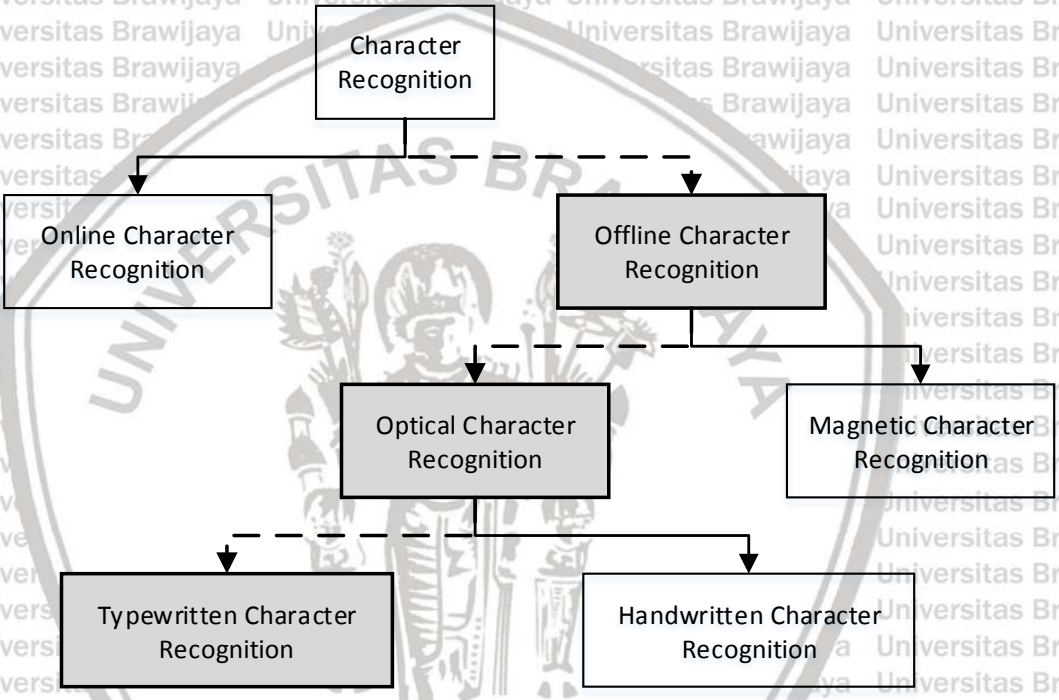
Lebih jauh, di sini akan membahas mengenai Hangul yang merupakan huruf Korea. Dalam penulisannya, membentuk kata Hangul lebih sulit dibandingkan dengan Latin karena untuk membuat satu kata, huruf Hangul tersusun dari dua dimensi, sedangkan huruf Latin langsung ditulis dari kiri dan kanan (Kyung-Won & Jin H., 2003).

Beberapa penelitian sudah dilakukan untuk mengenali karakter Hangul dalam suatu citra. Namun, kebanyakan penelitian tersebut hanya mengenali citra Hangul menjadi teks Hangul, tidak sampai pada merubah karakter Hangul menjadi Latin. Untuk itu, dalam penelitian ini akan mencoba melakukan pengenalan citra Hangul menjadi teks Latin.

Konversi dari citra Hangul menjadi teks Latin ini nantinya bisa digunakan sebagai pembelajaran bagaimana membaca huruf Hangul itu sendiri.

3.2 Kerangka Konsep

Pengenalan karakter dalam suatu citra bisa dilakukan secara *online* dan *offline*. Untuk pengenalan secara *offline*, terbagi menjadi pengenalan *magnetic* dan *optical*. Dalam *optical* jenis teks yang digunakan bisa berupa karakter tulisan tangan atau yang diketik. Jenis teks terakhir yang disebutkan akan digunakan sebagai data dalam penelitian ini. Untuk lebih jelasnya, kerangka konsep mengenai pengenalan karakter dalam penelitian ini tampak pada Gambar 3.1



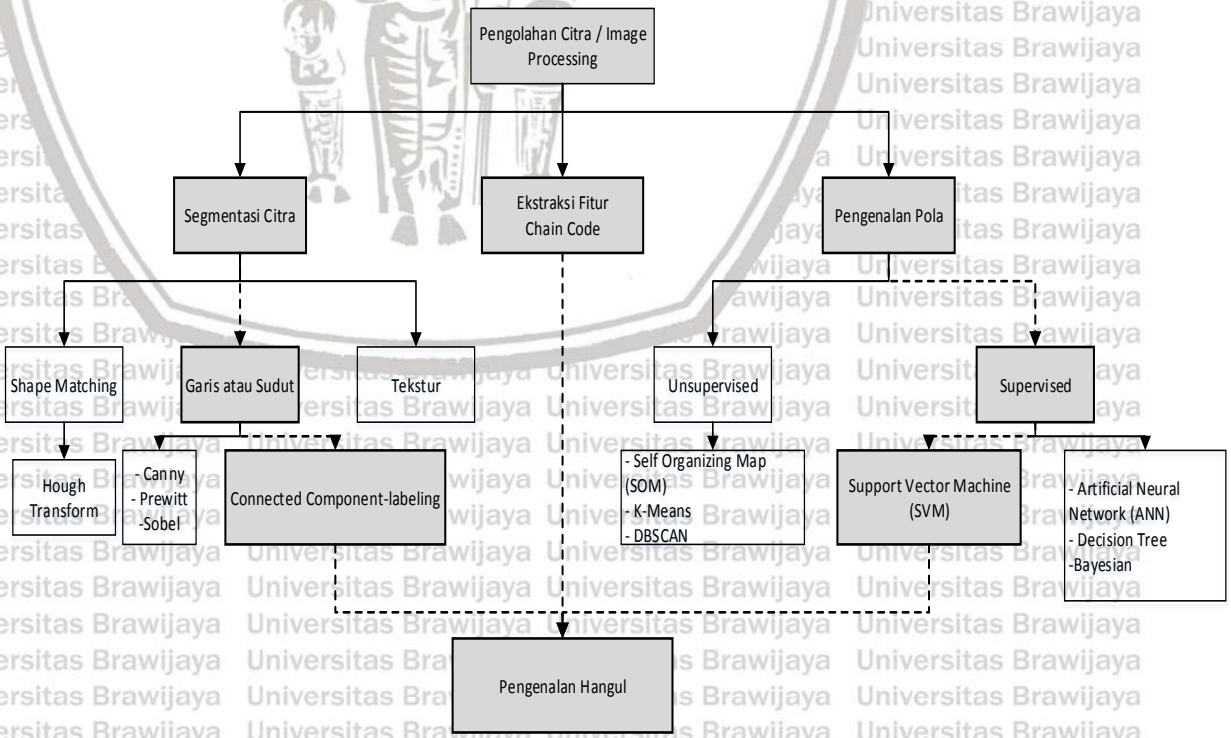
Gambar 3.1 Jenis Pengenalan Karakter (Dalbir & Singh, 2015)

Pengolahan citra memiliki beberapa cakupan. Di mana dalam penelitian ini masuk ke dalam pengenalan pola. Untuk pengenalan pola sendiri terbagi menjadi ke dalam 2 hal, yakni terbimbing (*supervised*) dan tidak terbimbing (*unsupervised*). *Supervised learning* merupakan pembelajaran terbimbing di mana proses pelatihan atau pembelajaran terhadap fungsi target memetakan setiap vector *x* ke dalam satu dari sejumlah label kelas *y* yang tersedia. Sedangkan *unsupervised learning* tidak ada *unsure* pelatihan atau pemberian label kelas, melainkan dalam pelaksanannya algoritma akan berjalan dengan sendirinya untuk mengelompokkan data. *Support Vector Machine* (SVM) merupakan salah satu metode *supervised learning*(Prasetyo, 2014).

Proses lain dalam pengolahan citra adalah segmentasi yang berguna untuk mengelompokkan citra menjadi beberapa *region.* Dalam penelitian ini segmentasi dilakukan untuk memisahkan huruf pada kata dengan menggunakan *connected component-labeling*. Hal ini berpengaruh pada proses ekstraksi fitur di mana sistem akan mengenali huruf-hurf tersebut sesuai urutan hasil segmentasi. Ekstraksi fitur yang akan digunakan di sini adalah *chain code/*kode rantai. Ekstraksi *chain code* menerapkan pencarian arah, di mana setiap arah diwakili oleh angka 0 sampai 7.

Setelah diketahui fitur masing-masing huruf, terakhir sistem melakukan pengenenalan di mana dalam hal ini menggunakan metode *Support Vector Machine* (SVM). Pada prinsipnya, metode SVM hanya bisa diterapkan pada 2 kelas. Sedangkan dalam pengenalan Hangul, terdapat 34 huruf yang berarti ada 34 kelas. Untuk mengatasi permasalahan SVM agar bisa diterapkan pada lebih dari 2 kelas, diperlukan suatu fungsi yang dikenal dengan fungsi *kernel*. Dalam penelitian ini, *kernel* yang akan digunakan adalah *linear, polynomial, radial basis function,* dan *laplacian*.

Gambaran teori dari pengolahan citra untuk pengenalan karakter Hangul ditunjukkan pada Gambar 3.2.



Gambar 3.2 Kerangka Konsep Penelitian

3.3 Studi Literatur

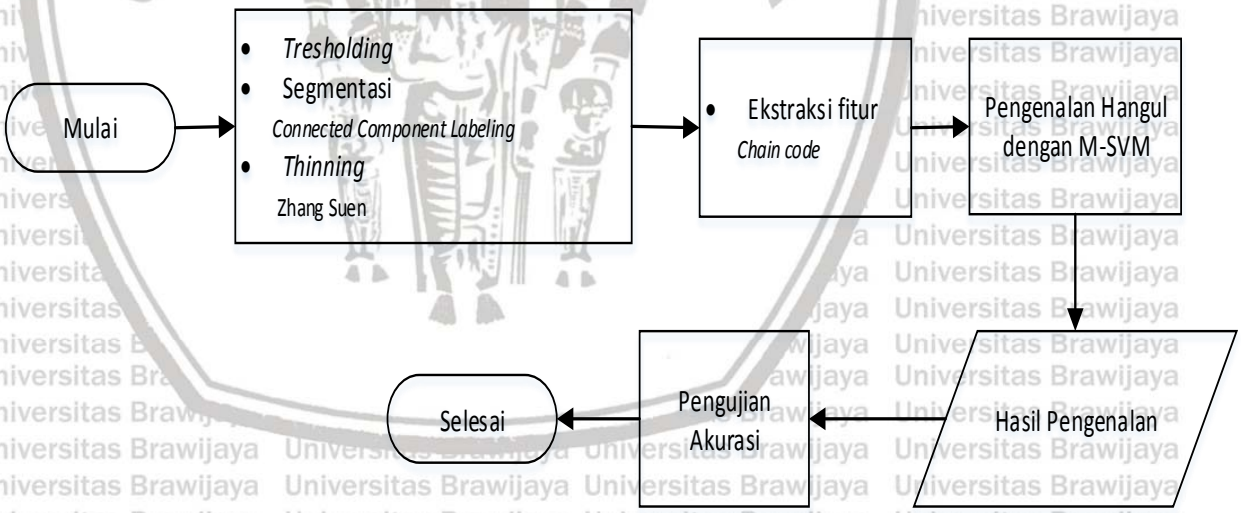
Adapun literature yang mendukung dalam penelitian ini adalah :

- Citra Hangul
- Pengolahan Citra Digital
- Segmentasi *Connected Component-labeling*
- *Thinning Zhang-Suen*
- Ektraksi Fitur *Chain Code*
- *Multiclass Support Vector Machine (M-SVM)*

Literatur tersebut diperoleh dari jurnal, buku, artikel dan dokumentasi penelitian.

3.4 Alur Langkah Penelitian

Alur langkah yang digunakan pada penelitian pengenalan karakter Hangul ditampilkan pada Gambar 3.3.



Gambar 3.3 Diagram Alir Langkah Penelitian

Dari diagram alir penelitian penenalan karakter Hangul pada Gambar 3.3, maka dapat dijelaskan langkah-langkah solusi masalah, yaitu:

1. Citra masukan dengan aras *grayscale* diubah dalam aras biner, sehingga bisa digunakan dalam proses segmentasi yang menggunakan *connected component labeling*. Kemudian sistem melakukan *thinning* dengan metode Zhang-Suen yang bertujuan mengurangi informsi citra untuk mempermudah pada proses ektraksi fitur.
2. Eksktraksi fitur dalam penelitian ini menggunakan *chain code* di mana sistem akan melakukan pencarian arah kode rantai sebagai fitur dari masing-masing citra.
3. Hasil ekstraksi fitur menjadi masukan proses pengenalan dengan menggunakan M-SVM.
4. Sistem melakukan perhitungan akurasi setiap *kernel* (*linear, polynomial, radial basis function*, dan *laplacian*).

3.5 Hipotesis

Segmentasi *connected component-labeling* mampu memberikan label pada citra sehingga proses pengambilan fitur dapat terurut sesuai susunan huruf dalam kata. Pengambilan fitur citra dilakukan dengan pencarian arah sesuai *chain code/* kode rantai yang ditetapkan. Fitur-fitur tersebut kemudian dinormalisasi agar diperoleh jumlah fitur yang sama sebelum dilanjutkan pada proses pengenalan menggunakan metode *Support Vector Machine* (SVM).

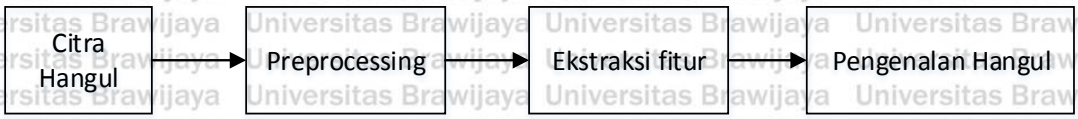
Fungsi *kernel* ditambahkan pada SVM agar bisa mengenali lebih 2 kelas. *Kernel* yang digunakan, yakni *linear, polynomial, radial basis function* dan *laplacian* mampu mengenali huruf dan kata Hangul dalam citra dengan tepat.

BAB IV
METODE PENELITIAN

4.1 Metode Penelitian

Metode penelitian yang diterapkan pada pengenalan karakter Hangul dimulai dari proses *input* citra yang berupa huruf atau kata Hangul. Selanjutnya melakukan *prepocessing*, di mana dalam hal ini terdiri dari 3 tahapan, pertama binerisasi atau *thresholding*, yakni merubah citra *grayscale* menjadi hitam putih. Kemudian segmentasi yang akan memisahkan *input* ke dalam *individual* huruf. Segmentasi dilakukan dengan menggunakan *connected component-labeling*. Tahap ketiga *thinning* yang bertujuan untuk mengurangi sejumlah informasi dalam sebuah pola. Dalam hal ini akan diterapkan algoritma paralel yang memiliki kinerja lebih cepat dibandingkan dengan algoritma *thinning* lain. Algoritma tersebut dikenal dengan Zhang-Suen (Haseena, et al, 2017). Dalam prosesnya, algoritma Zhang-Suen melakukan pengecekan beberapa kondisi terhadap ketetanggan piksel. Jika memenuhi kondisi-kondisi algoritma, maka piksel tersebut akan dihapus. Namun jika sebaliknya, maka piksel akan tetap dipertahankan.

Setelah *preprocessing*, langkah selanjutnya adalah ekstraksi fitur, yaitu mengambil fitur-fitur yang ada pada masing-masing huruf Hangul. Fitur-fitur tersebut didapat dari pencarian arah dengan menggunakan metode *chain code*. Agar bisa dilakukan proses pengenalan, jumlah fitur dari setiap huruf harus sama, untuk itu, perlu dilakukan normalisasi fitur. Setelah nilai tekstur diperoleh dan mendapakan jumlah yang sama, maka tahap terakhir adalah pengenalan dengan menggunakan metode *Support Vector Machine* (SVM). Berikut diagram alir dari penelitian pada Gambar 4.1



Gambar 4.1 Diagram Alir Metodologi Penelitian

4.2 *Input* Citra

Proses awal dari sistem ini adalah memasukkan data yang berupa citra Hangul berukuran 75×70 dalam format *bitmap*. Di mana dalam hal ini, huruf Hangul terdiri dari 34 huruf. Masing-masing huruf tersebut memiliki 15 pola. Jumlah keseluruhan data adalah 510 huruf. Data tersebut kemudian akan dibagi menjadi 3, yakni data A, B, dan C. Setiap data memiliki 5 pola atau 170 huruf. Data yang ada kemudian dibagi ke dalam 3 skenario data latih dan uji seperti yang tertera pada Tabel 4.1. Untuk tampilan keseluruhan data terdapat pada Lampiran 1.

Tabel 4.1 Skenario Data Pengenalan Huruf Hangul

Skenario Data	Data Latih	Data Uji
SD1	170 (Data A)	170 (Data C)
SD2	340 (Data A+B)	170 (Data C)
SD3	510 (Data A+B+C)	170 (Data C)

Untuk SD1 dan SD2, data uji menggunakan data baru yang belum dilatih. Sedangkan SD3, data uji memakai data yang pernah dilatih.

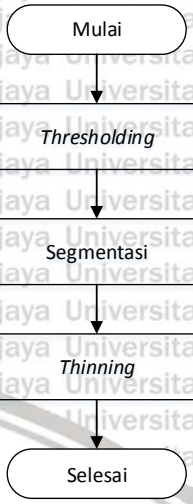
Untuk *input* kata, penulis membedakan berdasarkan tipe atau cara membentuk kata Hangul yang mana dalam hal ini menggunakan tipe 1 dan 2. Jumlah kata setiap tipe adalah 10 kata untuk tipe 1 dan 20 kata untuk tipe 2. Masing-masing kata memiliki 15 pola yang juga terbagi ke dalam Data A, B, dan C. Untuk skenario data pada proses pengujian sistem bisa dilihat pada Tabel 4.2.

Tabel 4.2 Data Latih dan Data Uji untuk Masing-masing Tipe Bentuk Hangul

Skenario Data	Tipe 1		Tipe 2	
	Data Latih	Data Uji	Data Latih	Data Uji
SD1	50 (Data A)	50 (Data C)	100 (Data A)	100 (Data C)
SD2	100 (Data A+B)	50 (Data C)	200 (Data A+B)	100 (Data C)
SD3	150 (Data A+B+C)	50 (Data C)	300 (Data A+B+C)	100 (Data C)

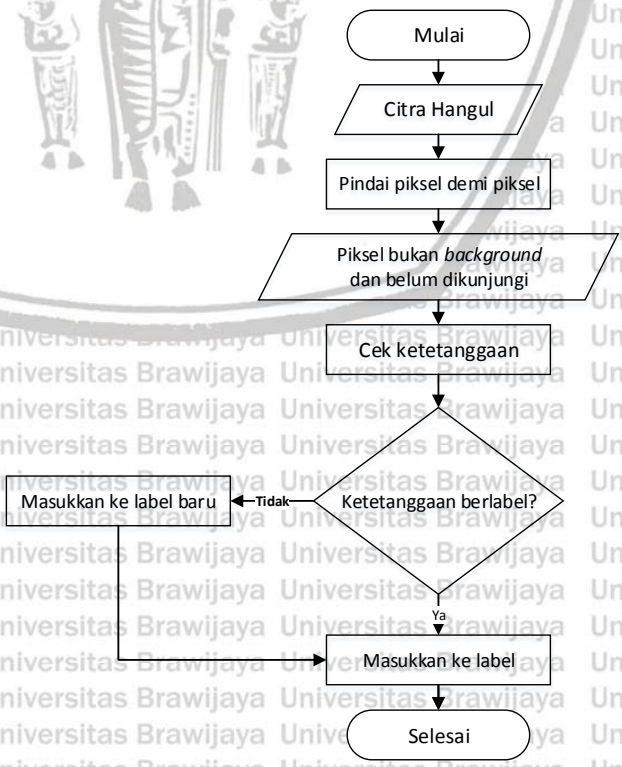
4.3 *Prepocessing*

Pada *prepocessing* terdapat 3 tahapan, yakni *thresholding* atau binerisasi, segemntasi, dan *thinning*. Diagram alir *prepocessing* bisa dilihat pada Gambar 4.2.



Gambar 4.2 Diagram Alir *Prepocessing*

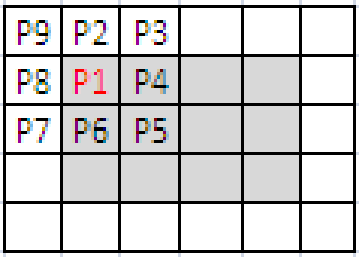
Proses awal dalam tahapan *preprocessing* adalah *tresholding*/binerisasi dengan nilai *threshold* 128. Tujuannya untuk mengubah citra Hangul *grayscale* menjadi citra biner (1 dan 0). Proses selanjutnya yaitu segmentasi. Segmentasi bertujuan untuk mengetahui goresan dari huruf maupun kata, sehingga mempermudah pengurutan pengambilan fitur. Proses segmentasi dilakukan menggunakan metode *connected component labeling* dengan 8 ketetanggan. Gambar 4.3 menampilkan diagram alir dari proses segmentasi.



Gambar 4.3 Diagram Alir Proses Segmentasi

Setelah segmentasi, selanjutnya sistem melakukan *thinning* atau penipisan untuk mengurangi sejumlah informasi dari pola. Hal ini berguna untuk

Algoritma Zhang Suen dipilih untuk proses *thinning* mengingat kinerjanya yang lebih cepat dengan algoritma *thinning* yang lain (Haseena & Clara, 2017). Algoritma Zhang Suen ini bekerja menggunakan matriks 3 × 3 dengan 8 ketetanggaan dari piksel (i,j), di mana piksel ketetanggaannya diasumsikan seperti pada Gambar 4.4. Piksel (i,j) merupakan P1 yang merupakan piksel yang akan dicek.



Gambar 4.4 Pengecekan piksel P1 menggunakan matriks 3x3 dengan 8 ketetanggan

P1 dicek dengan beberapa kondisi sebagai berikut (Zhang, T.Y. & Suen, C. Y., 1984):

(a) $\mathbf{2 \leq B(P_1) \leq 6}$ **(4-1)**

B(P1) merupakan jumlah tetangga-tetangga dari P1 yang bukan 0. Jadi, B(P1) = P2 + P3 + P4 + P5 + P6 + P7 + P8 + P9 harus lebih dari atau sama dengan 2 atau kurang dari atau sama dengan 6.

(b) $\mathbf{A(P_1) = 1}$ **(4-2)**

$A(P_1)$ merupakan jumlah transisi 0 ke 1 (*background* ke *foreground*) dalam urutan P2→P3, P3→P4, P4→P5, P5→P6, P6→P7, P7→P8, P8→P9, P9→P2.

Selanjutnya cek kondisi perkalian piksel pertama:

(c) $\boldsymbol{P2 \times P4 \times P8 = 0}$ **(4-3)**

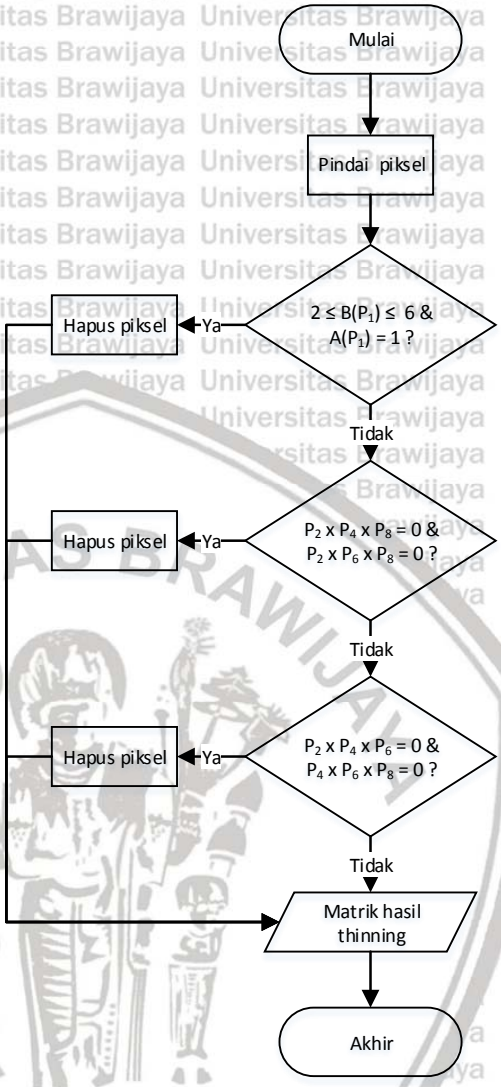
(d) $\boldsymbol{P2 \times P6 \times P8 = 0}$ **(4-4)**

Perkalian piksel kedua:

(e) $\boldsymbol{P2 \times P4 \times P6 = 0}$ **(4-5)**

(f) $\boldsymbol{P4 \times P6 \times P8 = 0}$ **(4-6)**

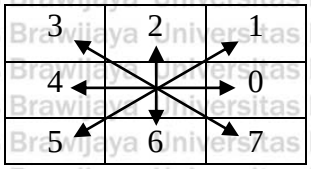
Diagram alir dari proses *thinning* dengan Zhang Suen tampak pada Gambar 4.5.



Gambar 4.5 Diagram Alir *Thinning* atau penipisan

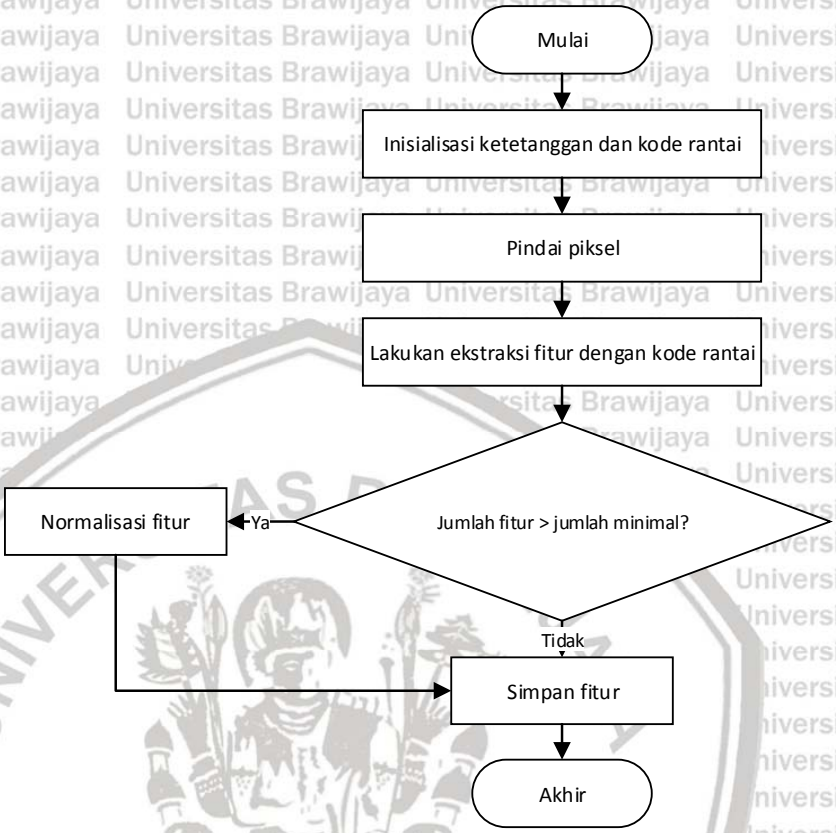
4.4 Ekstraksi Fitur

Proses ekstraksi fitur yang digunakan dalam penelitian ini adalah ekstraksi *chain code*. Sistem mengambil fitur-fitur dari setiap citra hasil *thinning* dengan pencarian arah. Gambar 4.6 menampilkan angka-angka yang mewakili pencarian arah.



Gambar 4.6 Arah kode rantai
Sumber: Sutoyo, dkk (2009)

Diagram alir dari proses ekstrasi fitur dengan *chain code* terdapat pada Gambar 4.7.



Gambar 4.7 Diagram Alir Ekstraksi Fitur

Normalisasi fitur dilakukan jika jumlah fitur melebihi jumlah minimal. Hal ini dilakukan agar fitur bisa diproses pada tahap selanjutnya. Persamaan (4-7) menampilkan bagaimana proses normalisasi fitur dilakukan.

$$Fitur_n = \frac{F_n}{N_n} \times N_{norm} \quad (4\text{-}7)$$

Di mana F_n merupakan frekuensi kemunculan fitur, N_n adalah jumlah fitur yang hendak dinormalisasi, dan N_{norm} adalah jumlah fitur minimal.

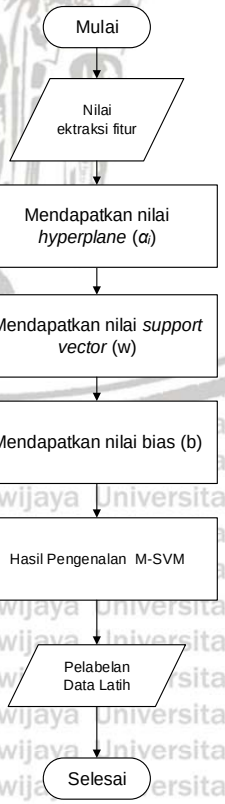
4.5 Pengenalan Menggunakan *Multiclass Support Vector Machine*

Pengenalan citra Hangul dibagi menjadi 34 kelas seusai dengan banyaknya huruf. Masukan pada citra sistem adalah citra huruf dan kata Hangul dalam format *bitmap*. Citra yang menjadi masukan akan diubah ke dalam *biner*, kemudian disegmentasi dengan *connected component labeling*. Selanjutnya dilakukan penipisan dengan menggunakan Zhang Suen.

Setelah proses *thinning* atau penipisan, citra diekstraksi menggunakan *chain code* untuk mengambil arah rantai. Fitur yang telah diperoleh kemudian diproses dengan algoritma *Multiclass Support Vector Machine*. Konsep *One-Against-One* digunakan untuk menentukan kelas data latih level 1 dan level selanjutnya. Hasil keluaran dari program berupa nilai *hyperplane,* nilai *support vector,* nilai *bias,* dan hasil pengenalan.

4.5.1 Proses Pelatihan *Multiclass Support Vector Machine*

Langkah pertama yang dilakukan pada proses pelatihan M-SVM adalah memberi masukan informasi berupa nilai ekstraksi fitur (huruf atau kata). Pada dasarnya proses *learning* dalam SVM yaitu mencari *support vector* untuk memperoleh *hyperplane* yang terbaik (Nugroho, 2008).*Hyperplane* yang paling baik diperoleh dengan memaksimalkan nilai *margin. Margin* (d) adalah jarak minimal antara *hyperplane* dan data latih. Untuk mendapatkan nilai *hyperplane* digunakan metode sekuensial (Vijaykumar,S., dan Wu, S., 1999). Perhitungn pada proses ini menggunakan Persamaan (2-15) sampai (2-21). Berikut disajikan diagram alir proses pelatihan M-SVM pada Gambar 4.8.

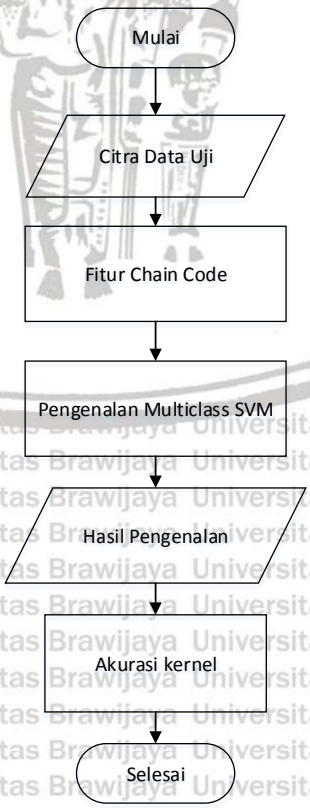


Gambar 4.8 Diagram Alir Proses Pelatihan *Multiclass Support Vector Machine*

4.5.2 Proses Pengujian *Multiclass Support Vector Machine*

Proses pengujian merupakan proses pengenalan yang menggunakan data uji huruf atau kata Hangul, di mana pada proses sebelumnya sudah diberikan label informasi pada proses pelatihan. Langkah pertama dalam pengujian adalah memasukkan citra data uji berupa hurf atau kata Hangul. Tahap selanjutnya adalah tahap *preprocessing* yaitu merubah *grayscale* menjadi *biner*, kemudian dilakukan segmentasi dengan *connected component labelling.* Selanjutnya dilanjutkan proses *thinning* atau penipisan dengan Zhang Suen. Sistem kemudian mengekstraksi fitur dengan pencarian arah atau kode rantai.

Proses selanjutnya adalah proses pengenalan menggunakan metode M-SVM untuk mengetahui apakah sistem berhasil mengenali citra Hangul menjadi teks Latin. Selain mendapatkan hasil pengenalan, keluaran lain yang didapatkan adalah nilai *support vector*, *bias*, dan nilai akurasi dari *kernel* dan iterasi yang diujikan. Berikut disajikan diagram alir proses pengujian pada Gambar 4.9.



Gambar 4.9 Diagram Alir Pengujian *Multiclass Support Vector Machine*

4.6 Uji *Kernel*

Pengujian sistem ini dilakukan untuk menguji keakuratan atau validasi dari perangkat lunak yang dibuat. Adapun proses pengujian yang dilakukan pada sistem ini adalah pengujian terhadap *kernel (linear, polynomial, RBF*, dan *laplacian*).

Pengujian terhadapa *kernel* dilakukan dengan alasan karena *kernel* memiliki pengaruh pada proses pencarian *hyperplane* secara optimal pada kasus SVM *non-linear* seperti pada penelitian ini. Pemilihan fungsi *kernel* yang tepat akan menentukan fitur baru (dimensi tinggi) dimana fungsi klasfikasi (*hyperplane*) akan dicari (Prasetyo, 2014).

BAB V
HASIL DAN PEMBAHASAN

5.1 *Input* Citra

Citra masukan dalam penelitian ini berupa citra *grayscale* huruf dan kata Hangul. Citra huruf Hangul terdiri dari 34 huruf, sedangkan kata terdapat 6 tipe atau cara dalam menulikannya. Masing-masing tipe memiliki jumlah berbeda. Citra yang dimasukkan dalam aplikasi berukuran 75 x 70 piksel. Contoh citra huruf dan kata Hangul tampak pada Gambar 5.1.

(a) (b)

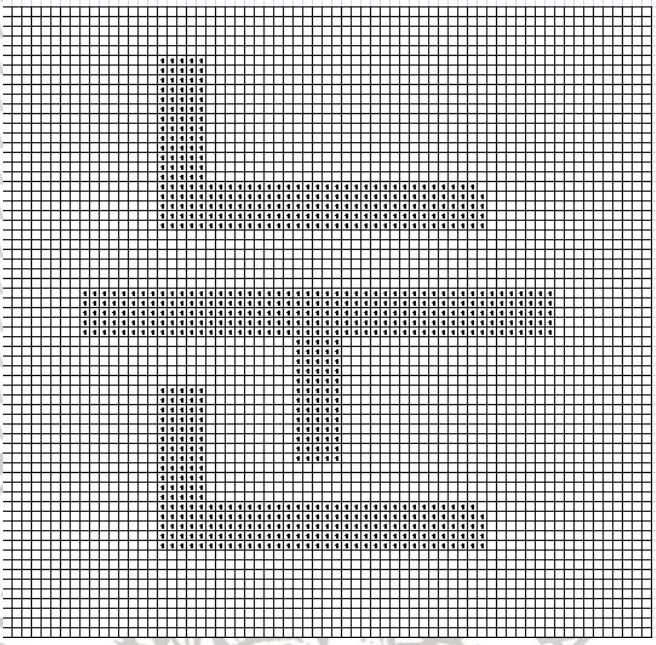
Gambar 5.1 Contoh Citra (a) Huruf L, (b) Kata Kim

5.2 *Prepocessing*

Setelah citra Hangul dimasukkan ke dalam sistem, maka proses selanjutnya adalah *preprocessing*. Pada proses ini, citra Hangul akan dikonversi ke dalam mode *biner*, kemudian disegmentasi. Segmentasi bertujuan untuk memberikan label pada huruf sehingga memudahkan pada proses pengurutan fitur.

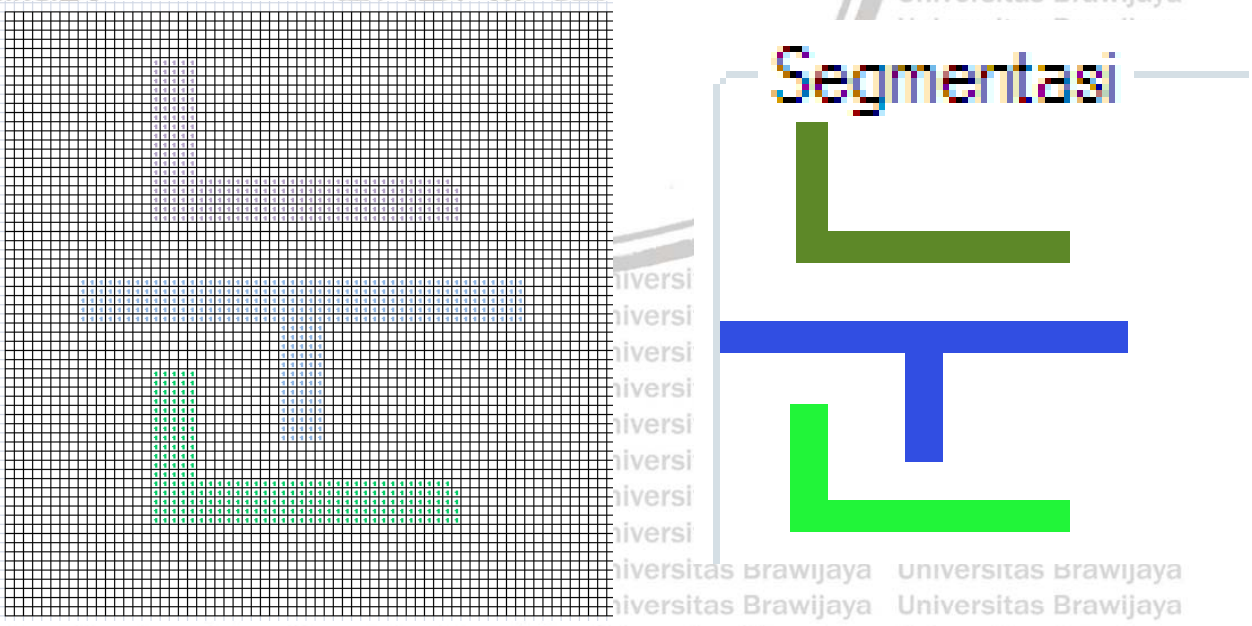
Proses segmentasi dimulai dengan memindai piksel demi piksel. Jika menemukan piksel bernilai satu, sistem akan mengecek apakah piksel tersebut berkaitan dengan label atau tidak. Jika tidak terkait dengan label manapun, maka sistem memberikan label baru. Selanjutnya, sistem memeriksa ketetanggaan dari piksel tersebut. Kemudian sistem memberikan warna secara *random* untuk membedakan setiap label.

Gambar 5.2 menampilkan citra kata Hangul dengan pikselnya. Tanda merah menunjukkan bahwa sistem mendeteksi piksel bernilai 1 yang nantinya akan dicek.



Gambar 5.2 Contoh kata NUN

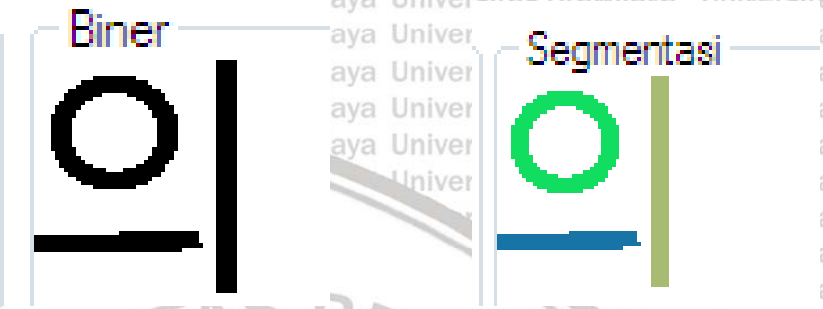
Dengan menggunakan 8 ketetanggaan, maka hasil segmentasi dari Gambar 5.2 akan nampak seperti Gambar 5.3.



(a) (b)

Gambar 5.3 (a) Ilustrasi piksel dari kata NUN, (b) Tampilan segmentasi kata NUN dari sistem

Pada Hangul, terdapat huruf yang penulisannya terdiri dari dua pola. Di sini, fungsi segmentasi bertujuan mengurutkan pola huruf tersebut agar tetap dikenali sebagai satu huruf. Contoh huruf yang memiliki lebih satu pola adalah UI. Gambar 5.4 menampilkan citra biner huruf UI dan hasil segmentasinya dalam sistem.

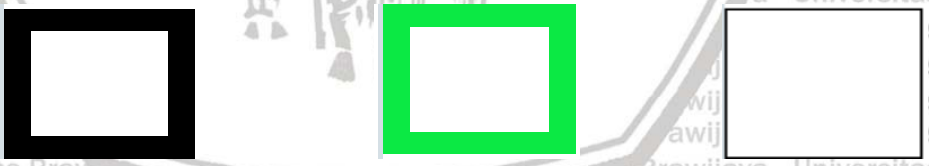


(a) (b)

Gambar 5.4 (a) Citra Biner Huruf UI, (b) Tampilan segmentasi huruf UI

Sebelum ke tahap fitur, sistem akan melakukan *thinning* atau penipisan pada hasil segmentasi. Hal ini untuk mengurangi jumlah informasi pada citra sehingga bisa mempercepat kinerja sistem. Metode *thinning* yang digunakan adalah Zhang Suen. Metode ini dipilih karena memiliki kinerja lebih cepat dibandingkan dengan algoritma *thinning* lain lain (A Review on an Efficient Iterative).

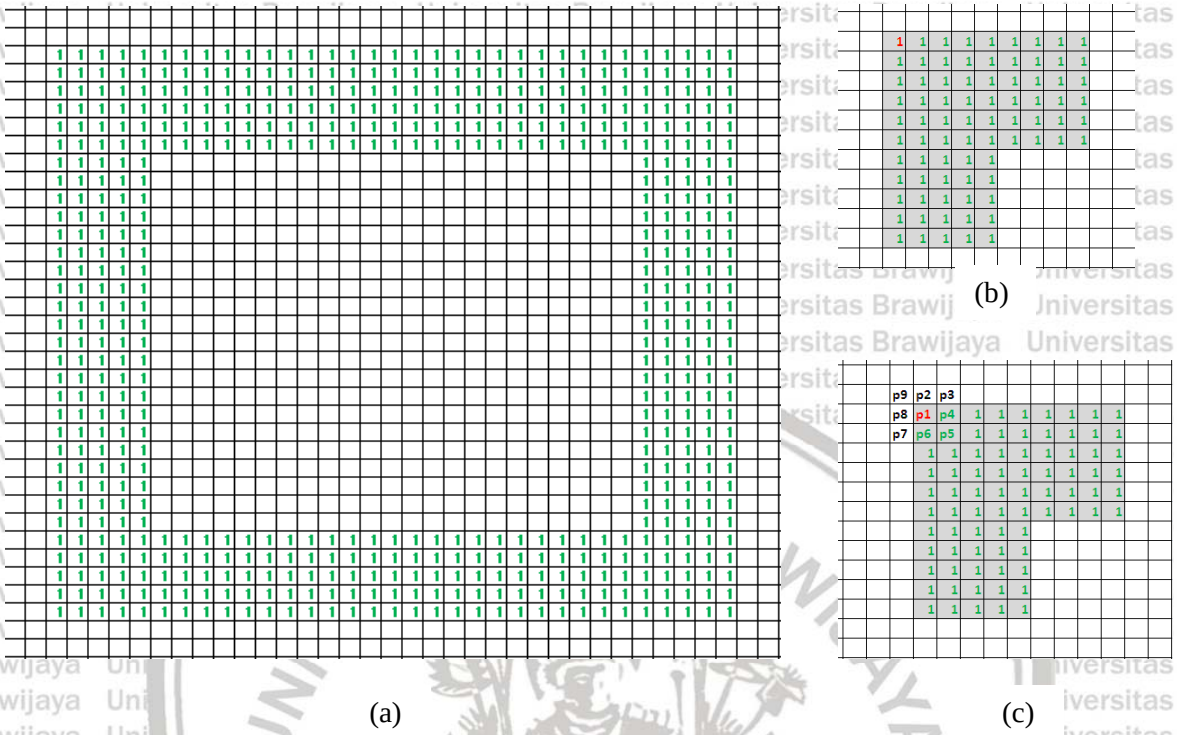
Keseluruhan gambaran dari tahap *preprocessing* ditampilkan pada Gambar 5.5.



(a) (b) (c)

Gambar 5.5 (a) Citra Biner M, (b) Segmentasi M, (c) *Thinning* M

Untuk proses *thinning,* sistem akan menghapus piksel yang memenuhi kondisi dari metode Zhang-Suen. Misal, di sini penulis akan menipiskan huruf M seperti pada Gambar 5.6.



(a) (b) (c)

Gambar 5.6 (a) Piksel huruf M, (b) Piksel yang diproses, (c) Ketetanggaan piksel

Piksel huruf M keseluruhan ditampilkan pada Gambar 5.5 (a), sedangkan di Gambar 5.5 (b) nampak sistem menemukan piksel yang hendak diproses. Dan Gambar 5.5 (c) merupakan gambaran 8 ketetanggaan dari piksel tersebut.

Langkah pertama proses thinning adalah sistem memindai piksel kemudian dicek apakah memenuhi kondisi berikut:

(a) $2 \leq B(P_1) \leq 6$ **(5-1)**

B(P1) merupakan jumlah ketetanggan P1 yang bukan 0. Berdasarkan Gambar 5.5, maka B(P1) = P4 + P5 + P6 = 1 + 1 + 1 = 3. Itu artinya, piksel tersebut memenuhi kondisi **$2 \leq B(P_1)=3 \leq 6$.**

(b) $A(P_1) = 1$ **(5-2)**

$A(P_1)$ merupakan jumlah transisi 0 ke 1 (*background* ke *foreground*) dalam urutan P2→P3, P3→P4, P4→P5, P5→P6, P6→P7, P7→P8, P8→P9, P9→P2. Terlihat di Gambar 5.5 terdapat 1 proses transisi 0 ke 1 pada P3→P4. Berarti, piksel tersebut memenuhi kondisi $A(P_1) = 1$.

Berhubung piksel memenuhi kondisi (a) dan (b), maka sistem akan melakukan penghapusan terhadap piksel tersebut. Namun jika piksel tersebut tidak memenuhi, maka lanjut mengecek kondisi berikut:

(c) $P2 \times P4 \times P8 = 0$**, dan** **(5-3)**

(d) $P2 \times P6 \times P8 = 0$**,** **(5-4)**

Selanjutnya sama, jika memenuhi kondisi (c) dan (d), piksel tersebut akan dihapus, sedangkan jika tidak memenuhi lakukan pengecekan:

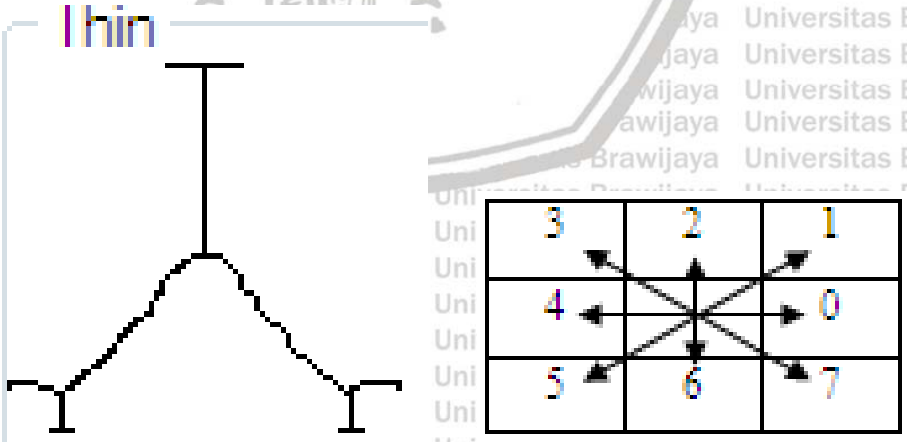
(a) $P2 \times P4 \times P6 = 0$**, dan** **(5-6)**

(b) $P4 \times P6 \times P8 = 0$ **(5-7)**

Jika pengecekan kondisi terakhir sistem tetap tidak memenuhi, maka sistem akan memindah piksel tersebut dalam citra baru. Dan itu berarti piksel tersebut merupakan piksel hasil *thinning* atau penipisan. Contoh hasil *thinning* ditampakkan pada Gambar 5.7 (a).

5.3 Hasil Ekstraksi Fitur *Chain Code*

Setelah tahap *prepocessing*, selanjutnya sistem mengekstraksi fitur citra menggunakan *chain code* atau pencarian arah. Tujuan dari proses ini adalah untuk memperoleh ciri atau fitur dari masing-masing huruf yang berjumlah 34. Contoh pencarian arah pada citra dengan menggunakan *chain code* terdapat pada Gambar 5.6



(a) (b)

Gambar 5.7 (a) Citra huruf CH hasil *thinning*, (b) Kode rantai

Pada Gambar 5.7 sistem melakukan pencarian arah fitur pada citra huruf CH dengan menggunakan kode rantai seperti pada Gambar 5.7 (b). Berdasarkan pencarian arah terebut, maka didapat fitur dari huruf S seperti berikut:

0;0;0;0;0;0;0;5;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;5;0;0;0;7;7;6;6;7;6;7;6;6;7;7;7;6;6;7;0;7;6;6;7;6;7;7;7;6;7;0;0;6;6;7;0;7;6;7;0;0;7;0;0;7;0;0;0;0;0;0;1;2;2;6;6;6;5;6;7;7;6;6;6;6;6;5;4;0;0;0;0;5;4;6;6;6;5;5;5;0;5;5;6;6;5;6;5;5;0;5;4;5;0;5;6;5;5;0;5;4;5;0;5;5;5;4;6;5;0;5;4;4;6;5;5;4;5;4;4;5;4;3;4;4;4;4;4;3;2;6;6;6;6;7;6;5;5;0;5;6;6;6;6;5;4;4;0;0;0;0;0;S;s_6.bmp

Pencarian fitur dilakukan bersamaan pada semua data. Agar bisa diproses pada tahapan selanjutnya, setiap citra harus memiliki jumlah fitur yang sama. Untuk memenuhi kondisi tersebut, maka dilakukan normalisasi fitur. Di sini sistem mencari jumlah yang paling sedikit dari keseluruhan citra. Kemudian melakukan normalisasi pada citra yang memiliki jumlah melebihi jumlah minimal dengan Persamaan (5-8) berikut:

$$Fitur_n = \frac{F_n}{N_n} \times N_{norm} \quad (5\text{-}8)$$

Perhitungan normalisasi huruf S sebagai berikut.

N_{notm}= 161

N_n = 184

Fitur 0 = $\frac{7}{184} \times 164 = 6.125 \approx 6$

Fitur 5 = $\frac{1}{184} \times 164 = 0.875 \approx 1$

Fitur 6 = $\frac{23}{184} \times 164 = 20.125 \approx 20$

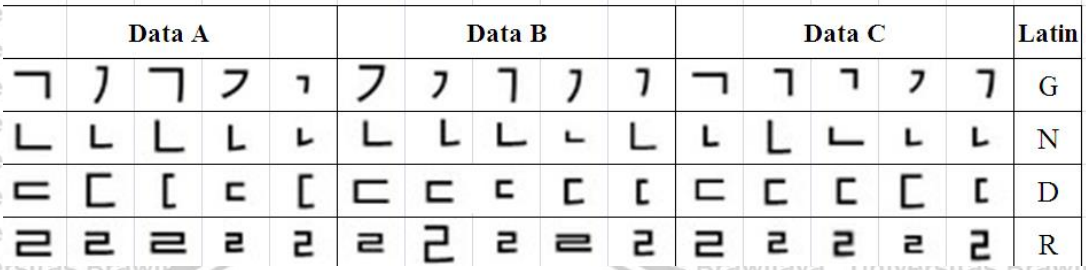
Setelah melakukan perhitungan dengan Persamaan (5-8), maka didapat hasil normalisi fitur sebagai berikut:

0;0;0;0;0;0;5;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;6;5;0;0;0;7;7;6;6;7;6;7;6;6;7;7;7;6;6;7;0;7;6;6;7;6;7;7;7;6;7;0;0;6;6;7;0;7;6;7;0;0;7;0;0;7;0;0;0;0;0;0;1;2;2;6;6;6;5;6;7;7;6;6;6;6;5;4;0;0;0;0;5;4;6;6;6;5;5;5;0;5;5;6;6;5;6;5;5;0;5;4;5;0;5;6;5;5;0;5;4;5;0;5;5;5;4;6;5;0;5;4;4;6;5;5;4;5;4;4;5;4;3;4;4;4;4;4;3;2;6;6;6;6;6;7;6;S;s_6.bmp

5.4 Hasil dan Analisis Proses Pengenalan M-SVM

Setelah mendapatkan fitur dari setiap huruf Hangul, maka proses selanjutnya adalah proses pengenalan berdasarkan informasi fitur dari data yang sudah dilatih. Untuk pengenalan huruf, terdapat 34 huruf yang memiliki 15 pola berbeda. Pola tersebut dibagi menjadi 3 data

skenario, yakni Data A, B, dan C. Jumlah keseluruhannya adalah 510 huruf. Tabel 5.1 menampilkan bagaimana data dibagi menjadi data latih dan data uji, sedangkan Gambar 5.8 mengkan bentuk dari polanya. Untuk tampilan pola keseluruhan data huruf bisa dilihat pada Lampiran 1.



Gambar 5.8 Contoh data huruf G, N, D, dan R dengan 15 pola

Tabel 5.1 Skenario Data untuk Sistem Pengenalan Huruf Hangul

Skenario Data	Data Latih	Data Uji
SD1	170 (Data A)	170 (Data C)
SD2	340 (Data A+B)	170 (Data C)
SD3	510 (Data A+B+C)	170 (Data C)

SD1 dan SD2 menggunakan data latih dan data uji yang berbeda. Sedangkan pada SD3 data uji merupakan data yang sudah dilatih. Akurasi yang diperoleh setiap skenario data tersebut untuk masing-masing *kernel* SVM ditampilkan pada Tabel 5.2. Terlihat bahwa semakin banyak data latih, maka tingkat akurasi semakin tinggi. Data hasil pengenalan huruf Hangul untuk semua data uji tertera di Lampiran 2.

Tabel 5.2 Hasil Akurasi Pengenalan Huruf Hangul

***Kernel* SVM**	**SD1**	**SD2**	**SD3**
Linear	88,82%%	94,11%	100%
Polynomial	89,41%	94,7%	100%
RBF	89,41%	94,7%	100%
Laplacian	84,11%	91,7%	100%

Kesalahan pengenalan sering terjadi pada huruf yang memiliki bentuk serupa, seperti 워 (WO) dikenali 위 (WI). Huruf lain yang nampak mirip adalah 우 (U) dengan 으 (EU), atau 얘(YAE) dengan 애 (AE). Keluaran sistem pengenalan Hangul ditampilkan pada Gambar 5.9.

Gambar 5.9 Tampilan Sistem Pengenalan Huruf Hangul *Kernel RBF*

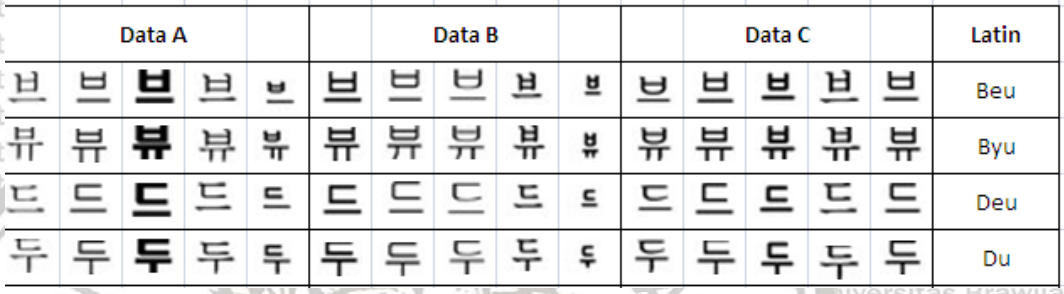
Untuk kata Hangul, sistem melakukan pengenalan pada tipe bentuk 1 dan 2. Jumlah kata setiap tipe adalah 10 kata untuk tipe 1 dan 20 kata untuk tipe 2. Masing-masing kata memiliki 15 pola yang juga terbagi ke dalam Data A, B, dan C. Perubahan pola tersebut didapat dari merubah jenis *font* yang mana dalam hal ini menggunakan *font Arial*, *Baekmuk Headline*, *Batang*, *Gaeul*, *Gulim*, *Dotum*, *DX 아기사랑 B*, *InterparkGothic*, *KoPub Batang*, *Malgun Gothic*, *Typo Donki Round*, *UnGraphic*, *UnJamoBatang*, *UnShinmun*, dan *UnTaza*. Untuk skenario data pada proses pengujian sistem bisa dilihat pada Tabel 5.3.

Tabel 5.3 Data Latih dan Uji masing-masing Tipe Bentuk Hangul

Skenario Data	**Tipe 1**		**Tipe 2**	
	Data Latih	**Data Uji**	**Data Latih**	**Data Uji**
SD1	50 (Data A)	50 (Data C)	100 (Data A)	100 (Data C)
SD2	100 (Data A+B)	50 (Data C)	200 (Data A+B)	100 (Data C)
SD3	150 (Data A+B+C)	50 (Data C)	300 (Data A+B+C)	100 (Data C)

Gambar 5.10 menampilkan contoh dari kata tipe 1 dan 2 dengan pola berbeda. Untuk keseluruhan data kata terdapat pada Lampiran 3.

(a)



(b)

Gambar 5.10 Contoh kata (a) Tipe 1 dan (b) Tipe 2 dengan 15 pola

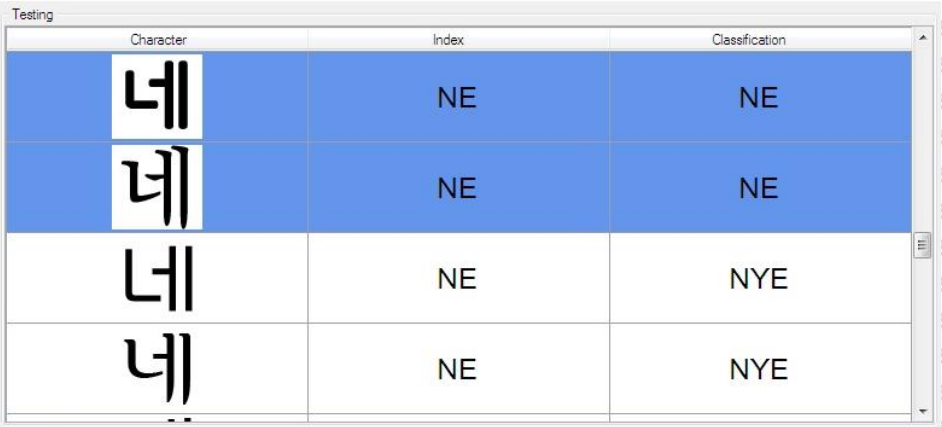
Akurasi dari masing-masing tipe dalam pengenalan kata Hangul pada setiap *kernel* SVM tertera pada Tabel 5.4. Sedangkan tampilan sistem penganalan kata Hangul terdapat pada Gambar 5.11.

Tabel 5.4 Hasil Akurasi Pengenalan Kata dari masing-masing Tipe

***Kernel* SVM**	**Tipe 1**			**Tipe 2**		
	SD1	SD2	SD3	SD1	SD2	SD3
Linear	60%	80%	100%	74%	80%	100%
Polynomial	62%	82%	100%	75%	81%	100%
RBF	62/%	64%	100%	65%	49%	100%
Laplacian	44%	46%	100%	45%	53%	100%

Kesalahan pengenalan tipe 1 terjadi pada kata yang memiliki penulisan hampir sama seperti 네 (NE) dikenali sebagai 녜 (NYE). Pada tipe 2 kesalahan ditemui pada huruf 두 (DU)

yang dikenali sebagai 드 (DEU) atau 그(GEU) dikenali 고(GO). Hasil keluaran semua kata dari masing-masing tipe terdapat pada Lampiran 4.



Gambar 5.11 Tampilan Sistem Pengenalan Kata Hangul Tipe 1 *Kernel Linear*

BAB VI
PENUTUP

6.1 Kesimpulan

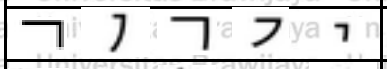
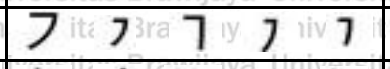
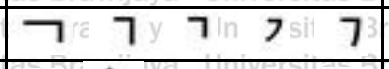
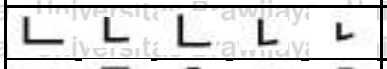
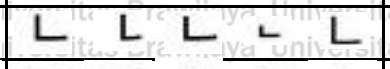
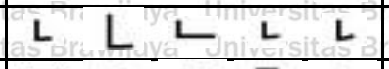
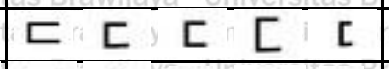
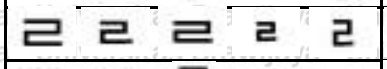
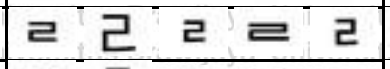
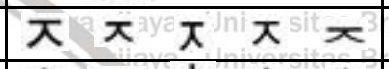
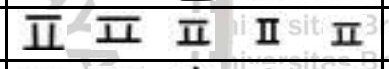
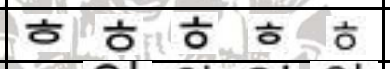
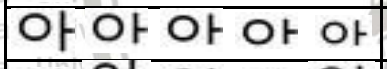
Dari pengujian yang dilakukan, dapat disimpulkan bahwa semakin banyak data latih maka tingkat akurasi yang diberikan oleh metode SVM semakin tinggi. Untuk pengenalan huruf Hangul dengan data uji yang berbeda dari data latih, akurasi tertinggi diperoleh ketika menggunakan *kernel polynomial* dan *RBF* sebesar 94,7% pada skenario data 2. Sedangkan untuk data uji yang pernah dilatih, semua *kernel* SVM (*linear*, *polynomial*, *RBF*, dan *laplacian*) memberikan akurasi 100%. Pada pengenalan kata, akurasi tertinggi mencapai 82% dengan menggunakan SVM *kernel polynomial* saat menggunakan tipe data 1 dan skenario data 2. Kesalahan pengenalan terjadi pada huruf atau kata yang memilki bentuk yang hampir sama. Contoh untuk huruf, 워 (WO) dikenali 위 (WI), sedangkan pada kata, 그(GEU) dikenali 고 (GO).

6.2 Saran

Untuk penelitian selanjutnya, pengambilan fitur dari citra bisa menggunakan metode fitur lain. Sedangkan pada proses pengenalan, bisa meneliti metode SVM dengan *kernel* berbeda. Pengenalan Hangul menjadi bentuk Latin ini juga bisa dikembangkan dengan menambahkan arti dari kata yang diuji.

Lampiran 1: Data Huruf Hangul

Data Citra Huruf Hangul (1)

Data A	Data B	Data C	Latin
			G
			N
			D
			R
			M
			B
			S
			J
			CH
			K
			T
			P
			H
			A
			YA
			EO
			YEO
			I
			AE
			YAE
			E
			YE
			O
			YO
			U
			YU
			EU
			WA

Data Citra Hangul (2)

Data A					Data B					Data C					Latin
왜	왜	왜	왜	왜	왜	왜	왜	왜	왜	왜	왜	왜	왜	왜	WAE
외	외	외	외	외	외	외	외	외	외	외	외	외	외	외	EO
워	워	워	워	워	워	워	워	워	워	워	워	워	워	워	WO
웨	웨	웨	웨	웨	웨	웨	웨	웨	웨	웨	웨	웨	웨	웨	WE
위	위	위	위	위	위	위	위	위	위	위	위	위	위	위	WI
의	의	의	의	의	의	의	의	의	의	의	의	의	의	의	UI

Lampiran 2: Hasil Pengenalan Huruf Hangul

Hasil Pengujian Pengenalan Huruf Hangul (1)

Keterangan: Li = *Kernel Linear* R = *Kernel RBF*
P = *Kernel Polynomial* La = *Kernel Laplacian*

NO.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 2			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
1	ㄱ	G_4	J	K	J	J	G	G	G	J	G	G	G	G
2	ㄱ	G_8	G	G	G	G	G	G	G	G	G	G	G	G
3	ㄱ	G_9	G	G	G	G	G	G	G	G	G	G	G	G
4	ㄱ	G_11	G	G	G	G	G	G	G	G	G	G	G	G
5	ㄱ	G_15	G	G	G	G	G	G	G	G	G	G	G	G
6	ㄴ	N_6	N	N	N	N	N	N	N	N	N	N	N	N
7	ㄴ	N_7	N	N	N	N	N	N	N	N	N	N	N	N
8	ㄴ	N_9	N	N	N	N	N	N	N	N	N	N	N	N
9	ㄴ	N_10	N	N	N	N	N	N	N	N	N	N	N	N
10	ㄴ	N_12	N	N	N	N	N	N	N	N	N	N	N	N
11	ㄷ	D_2	D	D	D	D	D	D	D	D	D	D	D	D
12	ㄷ	D_4	D	D	D	D	D	D	D	D	D	D	D	D
13	ㄷ	D_7	D	D	D	D	D	D	D	D	D	D	D	D
14	ㄷ	D_9	D	D	D	D	D	D	D	D	D	D	D	D
15	ㄷ	D_15	D	D	D	D	D	D	D	D	D	D	D	D
16	ㄹ	L_2	L	L	L	L	L	L	L	L	L	L	L	L
17	ㄹ	L_9	L	L	L	L	L	L	L	L	L	L	L	L
18	ㄹ	L_11	L	L	L	L	L	L	L	L	L	L	L	L
19	ㄹ	L_14	L	L	L	L	L	L	L	L	L	L	L	L
20	ㄹ	L_15	L	L	L	L	L	L	L	L	L	L	L	L
21	ㅁ	M_2	M	M	M	M	M	M	M	M	M	M	M	M
22	ㅁ	M_4	M	M	M	M	M	M	M	M	M	M	M	M
23	ㅁ	M_6	M	M	M	M	M	M	M	M	M	M	M	M
Jumlah 23 huruf														

Hasil Pengujian Pengenalan Huruf Hangul (2)

NO.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
24	ㅁ	M_9	M	M	M	M	M	M	M	M	M	M	M	M
25	ㅁ	M_12	M	M	M	M	M	M	M	M	M	M	M	M
26	ㅂ	B_1	B	B	B	B	B	B	B	B	B	B	B	B
27	ㅂ	B_5	B	B	B	B	B	B	B	B	B	B	B	B
28	ㅂ	B_8	B	B	B	B	B	B	B	B	B	B	B	B
29	ㅂ	B_12	B	B	B	B	B	B	B	B	B	B	B	B
30	ㅂ	B_14	B	B	B	B	B	B	B	B	B	B	B	B
31	ㅅ	S_2	S	S	S	S	S	S	S	S	S	S	S	S
32	ㅅ	S_7	S	S	S	S	S	S	S	S	S	S	S	S
33	ㅅ	S_11	S	S	S	S	S	S	S	S	S	S	S	S
34	ㅅ	S_14	S	S	S	S	S	S	S	S	S	S	S	S
35	ㅅ	S_15	S	S	S	S	S	S	S	S	S	S	S	S
36	ㅈ	J_3	J	J	J	J	K	K	J	J	J	J	J	J
37	ㅈ	J_8	K	K	J	J	K	K	J	J	J	J	J	J
38	ㅈ	J_11	K	K	J	J	J	J	J	J	J	J	J	J
39	ㅈ	J_14	K	K	J	J	J	J	J	J	J	J	J	J
40	ㅈ	J_15	K	K	J	J	K	K	J	J	J	J	J	J
41	ㅊ	CH_2	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
42	ㅊ	CH_4	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
43	ㅊ	CH_7	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
44	ㅊ	CH_9	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
45	ㅊ	CH_14	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
46	ㅋ	K_2	K	K	K	K	J	K	K	K	K	K	K	K
47	ㅋ	K_6	K	K	K	K	K	K	K	K	K	K	K	K
48	ㅋ	K_7	K	K	K	K	K	K	K	K	K	K	K	K
Jumlah 25 huruf														

Hasil Pengujian Pengenalan Huruf Hangul (3)

NO.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
49	ㅋ	K_8	K	K	K	K	K	K	K	K	K	K	K	K
50	ㅋ	K_14	K	K	K	K	K	K	K	K	K	K	K	K
51	ㅌ	T_2	T	T	T	T	T	T	T	T	T	T	T	T
52	ㅌ	T_5	T	T	T	T	T	T	T	T	T	T	T	T
53	ㅌ	T_9	T	T	T	T	T	T	T	T	T	T	T	T
54	ㅌ	T_12	T	T	T	T	T	T	T	T	T	T	T	T
55	ㅌ	T_14	T	T	T	T	T	T	T	T	T	T	T	T
56	ㅍ	P_3	O	O	O	O	P	P	P	P	P	P	P	P
57	ㅍ	P_5	YO	O	EU	O	P	P	P	P	P	P	P	P
58	ㅍ	P_9	YO	YO	EU	O	P	P	P	P	P	P	P	P
59	ㅍ	P_11	P	P	P	P	P	P	P	P	P	P	P	P
60	ㅍ	P_15	P	P	P	P	P	P	P	P	P	P	P	P
61	ㅎ	H_1	H	H	H	H	H	H	H	H	H	H	H	H
62	ㅎ	H_4	H	H	H	H	H	H	H	H	H	H	H	H
63	ㅎ	H_6	H	H	H	H	H	H	YAE	YAE	H	H	H	H
64	ㅎ	H_9	H	H	H	H	H	H	H	H	H	H	H	H
65	ㅎ	H_12	H	H	H	H	H	H	H	H	H	H	H	H
66	아	A_1	A	A	A	A	A	A	A	A	A	A	A	A
67	아	A_3	A	A	WO	WO	A	A	A	A	A	A	A	A
68	아	A_11	A	A	A	A	A	A	A	A	A	A	A	A
69	아	A_13	A	A	A	A	A	A	A	A	A	A	A	A
70	아	A_15	A	A	WA	WE	WA	WA	WA	WA	A	A	A	A
71	야	YA_5	YA	YA	A	WE	YA	YA	YA	YA	YA	YA	YA	YA
72	야	YA_6	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA
73	야	YA_9	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA

Jumlah 25 huruf

Hasil Pengujian Pengenalan Huruf Hangul (4)

NO.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
74	야	YA_10	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA
75	야	YA_13	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA	YA
76	어	EO_3	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO
77	어	EO_6	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO
78	어	EO_9	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO
79	어	EO_11	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO
80	어	EO_13	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO	EO
81	여	YEO_3	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO
82	여	YEO_6	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO
83	여	YEO_7	YEO	YEO	YEO	YEO	YEO	YEO	YEO	H	YEO	YEO	YEO	YEO
84	여	YEO_9	YEO	YEO	YEO	YEO	YEO	YEO	YEO	H	YEO	YEO	YEO	YEO
85	여	YEO_12	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO	YEO
86	이	I_6	I	I	I	I	I	I	I	I	I	I	I	I
87	이	I_9	YEO	AE	A	OE	I	I	I	I	I	I	I	I
88	이	I_10	WE	WE	WE	WE	I	I	I	I	I	I	I	I
89	이	I_11	I	I	I	I	I	I	I	I	I	I	I	I
90	이	I_13	I	I	I	I	I	I	I	I	I	I	I	I
91	애	AE_3	AE	AE	YAE	YAE	WAE	WAE	WAE	WAE	AE	AE	AE	AE
92	애	AE_7	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE
93	애	AE_9	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE
94	애	AE_11	AE	AE	AE	WAE	WAE	WAE	WAE	WAE	AE	AE	AE	AE
95	애	AE_14	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE	AE
96	얘	YAE_2	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE
97	얘	YAE_4	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE
98	얘	YAE_5	AE	AE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE
Jumlah 25 huruf														

Hasil Pengujian Pengenalan Huruf Hangul (5)

No.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
99	얘	YAE_6	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE
100	얘	YAE_10	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE	YAE
101	에	E_3	E	E	YE	YE	E	E	E	E	E	E	E	E
102	에	E_6	E	E	E	E	E	E	E	E	E	E	E	E
103	에	E_8	E	E	E	E	I	I	I	I	E	E	E	E
104	에	E_11	E	E	E	E	E	E	E	E	E	E	E	E
105	에	E_13	E	E	YE	YE	E	E	E	E	E	E	E	E
106	예	YE_1	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE
107	예	YE_2	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE
108	예	YE_5	YE	YE	YE	WE	YE	YE	WE	CH	YE	YE	YE	YE
109	예	YE_6	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE
110	예	YE_8	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE	YE
111	오	O_5	O	O	O	O	O	O	O	O	O	O	O	O
112	오	O_7	O	O	O	O	O	O	O	O	O	O	O	O
113	오	O_8	O	O	O	O	O	O	O	O	O	O	O	O
114	오	O_10	O	O	O	O	O	O	O	O	O	O	O	O
115	오	O-13	O	O	O	O	O	O	O	O	O	O	O	O
116	요	YO_3	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO
117	요	YO_4	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO
118	요	YO_6	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO
119	요	YO_10	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO
120	요	YO_12	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO	YO
121	우	U_5	U	U	U	U	U	U	U	U	U	U	U	U
122	우	U_6	U	U	U	U	U	U	U	U	U	U	U	U
123	우	U_9	U	EU	U	U	U	U	YU	YU	U	U	U	U

Jumlah 25 huruf

Hasil Pengujian Pengenalan Huruf Hangul (6)

NO.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
124	우	U_12	EU	EU	EU	EU	U	U	U	U	U	U	U	U
125	우	U_13	EU	EU	EU	EU	U	U	U	YU	U	U	U	U
126	유	YU_2	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU
127	유	YU_4	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU
128	유	YU_8	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU
129	유	YU_12	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU	YU
130	유	YU_14	YU	YU	YU	YU	YU	YU	U	U	YU	YU	YU	YU
131	으	EU_3	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU
132	으	EU_5	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU
133	으	EU_6	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU
134	으	EU_7	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU
135	으	EU_9	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU	EU
136	와	WA_2	WA	WA	WA	WA	A	A	WA	WA	WA	WA	WA	WA
137	와	WA_7	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA
138	와	WA_9	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA
139	와	WA_12	WA	WA	WA	WA	WE	WO	WA	WA	WA	WA	WA	WA
140	와	WA_14	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA	WA
141	왜	WAE_2	AE	AE	WAE	WO	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE
142	왜	WAE_6	WO	WO	AE	WO	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE
143	왜	WAE_10	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE
144	왜	WAE_13	AE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE
145	왜	WAE_15	AE	WAE	WAE	WO	WAE	WAE	WAE	WAE	WAE	WAE	WAE	WAE
146	외	OE_4	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE
147	외	OE_6	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE
148	외	OE_9	OE	OE	OE	OE	OE	OE	OE	I	OE	OE	OE	OE
Jumlah 25 huruf														

Hasil Pengujian Pengenalan Huruf Hangul (7)

NO.	Hangul	Latin	Hasil Pengujian											
			Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
149	외	OE_11	AE	AE	YAE	OE	OE	OE	OE	OE	OE	OE	OE	OE
150	외	OE_12	AE	AE	YAE	OE	OE	OE	OE	OE	OE	OE	OE	OE
151	워	WO_1	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
152	워	WO_5	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
153	워	WO_7	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
154	워	WO_9	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
155	워	WO_11	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
156	웨	WE_3	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE
157	웨	WE_8	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE
158	웨	WE_10	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE
159	웨	WE_11	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE
160	웨	WE_15	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE	WE
161	위	WI_4	WI	WI	WI	WO	WI	WI	WI	WI	WI	WI	WI	WI
162	위	WI_7	WI	WI	WI	WO	WI	WI	WI	WI	WI	WI	WI	WI
163	위	WI_8	WI	WI	WI	WO	WI	WI	WI	WI	WI	WI	WI	WI
164	위	WI_11	WI	WI	WI	WO	WI	WI	WI	WI	WI	WI	WI	WI
165	위	WI_15	WI	WI	WI	WO	WI	WI	WI	WI	WI	WI	WI	WI
166	의	UI_1	UI	UI	UI	WO	WI	UI	WI	WI	UI	UI	UI	UI
167	의	UI_2	UI	UI	UI	UI	UI	UI	UI	UI	UI	UI	UI	UI
168	의	UI_7	UI	UI	WI	WO	UI	UI	UI	UI	UI	UI	UI	UI
169	의	UI_11	UI	UI	UI	UI	UI	UI	UI	UI	UI	UI	UI	UI
170	의	UI_13	UI	UI	UI	WO	UI	UI	UI	UI	UI	UI	UI	UI

Jumlah 22 huruf

Lampiran 3: Data Kata Hangul

Kata Hangul Tipe 1

Data A	Data B	Data C	Latin
가 가 **가** 가 가	가 가 가 가 가	가 가 **가** 가 가	GA
개 개 **개** 개 개	개 개 개 개 개	개 개 **개** 개 개	GAE
너 너 **너** 너 너	너 너 너 너 너	너 너 **너** 너 너	NEO
네 네 **네** 네 네	네 네 네 네 네	네 네 **네** 네 네	NE
녀 녀 **녀** 녀 녀	녀 녀 녀 녀 녀	녀 녀 **녀** 녀 녀	NYEO
녜 녜 **녜** 녜 녜	녜 녜 녜 녜 녜	녜 녜 **녜** 녜 녜	NYE
니 니 **니** 니 니	니 니 니 니 니	니 니 **니** 니 니	NI
러 러 **러** 러 러	러 러 러 러 러	러 러 **러** 러 러	LEO
레 레 **레** 레 레	레 레 레 레 레	레 레 **레** 레 레	LE
미 미 **미** 미 미	미 미 미 미 미	미 미 **미** 미 미	MI

Lampiran 4: Hasil Pengenalan Kata

Hasil Pengujian Pengenalan Kata Tipe 1 (1)

NO.	Hangul	Latin	Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
1	가	GA_4	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA
2	가	GA_6	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA
3	가	GA_25	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA
4	가	GA_26	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA	GA
5	가	GA_28	GAE	GAE	GAE	LE	GA	GA	GA	GA	GA	GA	GA	GA
6	개	GAE_4	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE
7	개	GAE_6	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE
8	개	GAE_25	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE
9	개	GAE_26	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE
10	개	GAE_28	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE	GAE
11	너	NEO_4	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO
12	너	NEO_6	NEO	NEO	NEO	NYE	NEO	NEO	LE	GAE	NEO	NEO	NEO	NEO
13	너	NEO_25	NEO	NEO	NEO	NYE	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO
14	너	NEO_26	NEO	NEO	NEO	GAE	NEO	NEO	LE	GAE	NEO	NEO	NEO	NEO
15	너	NEO_28	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO	NEO
16	네	NE_4	NE	NE	NE	NYE	NE	NE	NE	NYE	NE	NE	NE	NE
17	네	NE_6	NE	NE	NE	LE	NE	NE	LE	NYE	NE	NE	NE	NE
18	네	NE_25	NYE	NYE	NYE	NYE	NE	NE	NYE	NYE	NE	NE	NE	NE
19	네	NE_26	NYE	NE	NE	NYE	NE	NE	LE	NYE	NE	NE	NE	NE
20	네	NE_28	LE	LE	LE	NYE	NYE	NYE	NE	NYE	NE	NE	NE	NE
21	녀	NYEO_4	NYEO	NYEO	NYEO	NYEO	NI	NI	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO
22	녀	NYEO_6	NYEO	NYEO	NYEO	LE	NYEO	NYEO	LE	GA	NYEO	NYEO	NYEO	NYEO
23	녀	NYEO_25	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO
		Jumlah 23 kata												

Hasil Pengujian Kata Tipe 1 (2)

	Hangul	Latin	Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P(81%)	R	La	Li	P	R	La
24	녀	NYEO_26	NYEO	NYEO	NYEO	LE	NYEO	NYEO	LE	GAE	NYEO	NYEO	NYEO	NYEO
25	녀	NYEO_28	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO	NYEO
26	녜	NYE_4	NE	NE	NE	NE	NE	NE	NYE	NYE	NYE	NYE	NYE	NYE
27	녜	NYE_6	NYE	NYE	NYE	NYE	NYE	NYE	LE	NYE	NYE	NYE	NYE	NYE
28	녜	NYE_25	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE
29	녜	NYE_26	NE	NE	NE	NYE	NE	NE	LE	GA	NYE	NYE	NYE	NYE
30	녜	NYE_28	LE	LE	LE	LE	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE
31	니	NI_4	NEO	NEO	NEO	NI	NEO	NYE	NI	NI	NI	NI	NI	NI
32	니	NI_6	NI	NI	NI	MI	NI	NI	NI	NEO	NI	NI	NI	NI
33	니	NI_25	NEO	NEO	NEO	NEO	NI	NI	NEO	NI	NI	NI	NI	NI
34	니	NI_26	NI	NI	NI	LE	NI	NI	LE	NEO	NI	NI	NI	NI
35	니	NI_28	NI	NI	NI	NEO	NI	NI	NI	NI	NI	NI	NI	NI
36	러	LEO_4	MI	MI	MI	MI	LE	LE	LE	GAE	LEO	LEO	LEO	LEO
37	러	LEO_6	GAE	GAE	GAE	GAE	LE	LE	LE	GAE	LEO	LEO	LEO	LEO
38	러	LEO_25	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO
39	러	LEO_26	NYE	NYE	NYE	MI	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO
40	러	LEO_28	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO
41	레	LE_4	NYE	GAE	GAE	NYE	LE	LEO	LEO	GAE	LE	LE	LE	LE
42	레	LE_6	NYE	NYE	NYE	NYE	NYE	NYE	NYE	NYE	LE	LE	LE	LE
43	레	LE_25	NYE	NYE	NYE	NYE	LE	LE	LE	GAE	LE	LE	LE	LE
44	레	LE_26	NE	NE	NE	NYE	NE	MI	MI	NYE	LE	LE	LE	LE
45	레	LE_28	LEO	LE	LE	LE	LE	LE	LE	NYE	LE	LE	LE	LE
46	미	MI_4	LEO	LEO	LEO	LEO	MI	MI	LE	GA	MI	MI	MI	MI

Jumlah 23 kata

Hasil Pengujian Kata Tipe 1 (3)

NO.	Hangul	Latin	Skenario Data 1				Skenario Data 2				Skenario Data 3			
			Li	P	R	La	Li	P	R	La	Li	P	R	La
47	미	MI_6	LE	GA	GA	LE	MI	MI	GA	GA	MI	MI	MI	MI
48	이	MI_25	LEO	LEO	LEO	LEO	MI	MI	MI	LEO	MI	MI	MI	MI
49	미	MI_26	LEO	LEO	LEO	LEO	LEO	LEO	LEO	LEO	MI	MI	MI	MI
50	이	MI_28	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI
Jumlah 4 kata														