Pengenalan Tulisan Tangan Karakter Jepang Menggunakan Library Tesseract Pada Android

Rony Try Haryanto[1], Reza Mahardityawarman[2], Kusnaedi[3] and Dyas Yudi Priyanggodo[4]

***Abstract*— Lately, digital image processing in many developed countries into fields cultivated by many researchers as attractive to apply to various activities, both analysis and production activities. One of the branches in the digital image is pattern recognition. This study uses Tesseract as a tool to recognize patterns of Japanese letter. This research was conducted to determine how much Tesseract is able to recognize an Japanese text and handwritten text. Common Japanese writing system are Hiragana and Katagana. The objective of the paper is to recognize handwritten samples of Japanese using Tesseract open source Optical Character Recognition (OCR) engine. Tesseract is trained with data samples of different persons to generate one user-independent language model, representing the handwritten Japanese digit-set.**

***Index Terms*— Japanese OCR, tesseract, digital image processing.**

I. PENDAHULUAN

Jepang sebagai salah satu negeri maju di Asia mempunyai daya tarik tersendiri bagi para pencari kerja yang berasal dari luar Jepang. Namun salah satu kendala bagi para pencari kerja ini adalah dari segi bahasa, yaitu bangsa Jepang tidak menggunakan aksara Latin dalam kehidupan sehari-hari melainkan menggunakan aksara yang berasal dari tulisan bahasa Cina yang diperkenalkan pada abad keempat Masehi. Saat ini tulisan Jepang terbagi menjadi tiga kategori yaitu Kanji, Hiragana dan Katakana.

Mempelajari tulisan dalam bahasa Jepang diperlukan latihan untuk dapat menghafal aksara-aksara tersebut. Dalam rangka membantu menghafal aksara Jepang, maka pada penelitian kali ini akan dibuat sistem yang dapat mengecek apakah penulisan sudah sesuai dengan aksara Jepang. Pengenalan tulisan tangan membutuhkan teknologi pengenalan citra digital dengan teknik Optical Character Recognition (OCR). OCR sendiri adalah teknik untuk mengubah teks non digital menjadi teks digital atau secara harfiah dapat diartikan sebagai pengenalan karakter optik. Selain pengenalan tulisan tangan, aplikasi harus dapat tersedia dengan cepat mudah digunakan, maka sistem akan dibuat pada perangkat mobile berbasis Android.

II. DASAR TEORI

A. *Hiragana*

Hiragana adalah huruf Bahasa Jepang asli yang dibuat oleh orang jepang. Huruf ini mempunyai fungsi sebagai kata-kata asli Bahasa Jepang yang bukan kata serapan. Untuk daftar aksara dapat dilihat pada Gambar 1.

ひらがな Hiragana									
Seion	あ	a	い	i	う	u	え	e	お	o
	か	ka	き	ki	く	ku	け	ke	こ	ko
	さ	sa	し	shi	す	su	せ	se	そ	so
	た	ta	ち	chi	つ	tsu	て	te	と	to
	な	na	に	ni	ぬ	nu	ね	ne	の	no
	は	ha	ひ	hi	ふ	fu	へ	he	ほ	ho
	ま	ma	み	mi	む	mu	め	me	も	mo
	や	ya			ゆ	yu			よ	yo
	ら	ra	り	ri	る	ru	れ	re	ろ	ro
	わ	wa							を	wo
	ん	n								

Gambar 1. Daftar Aksara Hiragana

B. *Katagana*

Huruf Katakana biasa dipakai untuk menulis kata serapan dari bahasa asing. Sebagaimana alfabet, huruf Katana dan Hiragana hanya mewakili satu bunyi tanpa arti. Walaupun kalimat dalam bahasa Jepang biasa terdiri dari Hiragana, Katakana dan Kanji, tetapi bisa juga cuma ditulis dalam Hiragana dan Katakana. Untuk daftar aksara dapat dilihat pada Gambar 2.

かたかな Katakana									
Seion	ア	a	イ	i	ウ	u	エ	e	オ	o
	カ	ka	キ	ki	ク	ku	ケ	ke	コ	ko
	サ	sa	シ	shi	ス	su	セ	se	ソ	so
	タ	ta	チ	chi	ツ	tsu	テ	te	ト	to
	ナ	na	ニ	ni	ヌ	nu	ネ	ne	ノ	no
	ハ	ha	ヒ	hi	フ	fu	ヘ	he	ホ	ho
	マ	ma	ミ	mi	ム	mu	メ	me	モ	mo
	ヤ	ya			ユ	yu			ヨ	yo
	ラ	ra	リ	ri	ル	ru	レ	re	ロ	ro
	ワ	wa							ヲ	wo
	ン	n								

Gambar 2. Daftar Aksara Katagana

C. *Optical Character Recognition (OCR)*

Optical character recognition (OCR) adalah sebuah sistem komputer yang dapat membaca huruf, baik yang

berasal dari sebuah pencetak (printer atau mesin ketik) maupun yang berasal dari tu lisan tangan. OCR adalah aplikasi yang menerjemahkan gambar karakter (image character) menjadi bentuk teks dengan cara menyesuaikan pola karakter per baris dengan pola yang telah tersimpan dalam database aplikasi. Hasil dari proses OCR adalah berupa teks sesuai dengan gambar output scanner dimana tingkat keakuratan penerjemahan karakter tergantung dari tingkat kejelasan gambar dan metode yang digunakan[xxx].

D. Tesseract Engine

Proses pengenalan karakter dalam penelitian ini menggunakan library Tesseract. Menurut Smith (2007), Tesseract adalah suatu engine Optical Character Recognition. Engine ini pertama kali dikembangkan oleh Hewlett-Packard pada tahun 1985. Pada tahun 2005, Tesseract dirilis sebagai open source oleh Hewlett-Packard dan UNLV. Semenjak tahun 2006, pengembangan Tesseract disponsori oleh Google dan dirilis dengan lisensi apache versi 2.0. Versi stabil Tesseract pada saat ini adalah 3.01.

Berikut ini algoritma dari Tesseract engine :

1. Image input

   Gambar berwarna atau grayscale diberikan sebagai input. Tesseract menerima file dengan ekstensi .tiff dan .bmp secara native namun terdapat plug-in untuk memproses gambar dengan format kompresi lainnya. Input yang ideal bagi Tesseract adalah gambar yang flat.

2. Adaptive Thresholding

   Melakukan reduksi dari grayscale image ke binary image. Algoritma mengasumsikan gambar terdiri dari foreground pixel dan background pixel. Kemudian, menghitung threshold yang optimal untuk memisahkan kedua pixel tersebut.

3. Connected-Component labeling

   Tesseract menelusuri pixel dalam gambar, mengidentifikasi foreground pixel, dan ditandai sebagai blob atau potential character.

4. Line Finding Algorithm

   Garis dari teks ditemukan dari hasil analisa space gambar yang berdekatan dengan potential character. Algoritma ini mencari lokasi yang memiliki pixel kurang dari threshold tertentu. Hasilnya berupa area yang ditandai sebagai potential line.

5. Baseline fitting algorithm

   Setelah setiap garis dari teks ditemukan, Tesseract memeriksa garis teks untuk memperkirakan tinggi teks. Proses ini merupakan langkah awal dalam mengenali karakter.

6. Fixed Pitch Detention

   Tesseract memperkirakan lebar karakter. Nilai ini merupakan nilai incremental yang akan digunakan oleh Tesseract dalam mengekstrak karakter demi karakter.

7. Non-fixed pitch spacing delimiting

   Karakter yang tidak seragam dengan lingkungan disekitar karakter tersebut akan diklasifikasi ulang untuk diproses secara terpisah dari keseluruhan gambar.

8. Word Recognition

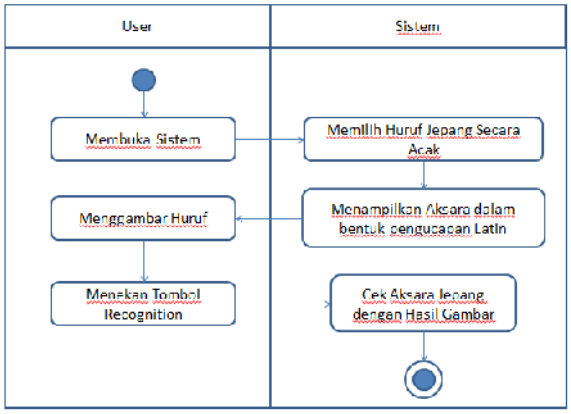
   Setelah menemukan seluruh possible character dan possible line dalam gambar, Tesseract melakukan word recognition dengan menelusuri garis. Kata kemudian dikirim kepada contextual dan syntactical analyzer untuk meningkatkan tingkat akurasi.

E. Android

Android merupakan salah satu sistem operasi perangkat mobile yang tergolong masih baru dan sangat berkembang. Android bersifat open source dan pertama kali di rilis oleh Google pada tahun 2009 dan sejak saat itu Sistem Operasi Android terus berkembang dengan pesat dan berhasil mendapatkan perhatian dari jutaan mobile phone user dan mobile developer. Hingga saat ini telah banyak vendor dari perangkat mobile yang telah menggunakan Sistem Operasi Android pada produk-produk mereka. Untuk memenuhi kebutuhan pengembangan, Google bersama dengan OHA merilis paket Android SDK (*Software Development Kit*) dan ADT (*Android Development Tools*) untuk mengembangkan aplikasi Android pada perangkat mobile. Android SDK sendiri terdiri dari sistem operasi, middleware dan aplikasi utama untuk perangkat mobile. Bahasa pemrograman yang digunakan pada Android adalah bahasa pemrograman Java yang telah diberikan fungsi-fungsi khusus untuk pengembangan Android sendiri. Dengan Android SDK dan ADT, developer bisa bebas berkreasi dalam menciptakan aplikasi-aplikasi yang nantinya bisa dipasarkan dan digunakan oleh banyak Android user.

III. CARA KERJA SISTEM

Secara umum, cara kerja sistem adalah sistem menampilkan pengucapan aksara yang akan dijadikan soal kepada pengguna. Kemudian pengguna diharuskan menebak penulisan dalam aksara Jepang dengan cara menggambarkan pada perangkat Android. Lalu sistem akan mencocokkan kedua aksara Jepang yang dimaksud dengan aksara yang ditulis oleh pengguna. Berikut skema diagram aplikasi (Gambar 3):



Gambar 3. Diagram Aplikasi

Detail dari proses yang ada pada sistem dijabarkan sebagai berikut :

A. Proses Membuka Aplikasi

Pada proses ini pengguna diharuskan melakukan instalasi sistem pada perangkat Andoid terlebih dahulu, kemudian membuka menu untuk memulai proses menebak aksara Jepang dengan cara menggambarnya pada perangkat menggunakan jari.

B. Proses Menampilkan Soal

Setelah pengguna membuka menu, maka sistem akan mengambil aksara yang akan dijadikan sebagai soal kepada pengguna secara acak. Kemudian sistem akan mencari kata pengucapan aksara Jepang tersebut dalam bahasa Latin. Misalnya, sistem mengambil aksara カ sebagai soal maka sistem akan menampilkan pengucapan aksara tersebut yaitu "**ka**" dan menyembunyikan aksara dalam bahasa Jepang, sehingga pengguna harus menebak aksara tersebut.

C. Proses Menebak

Pada layar aplikasi akan terdapat huruf Latin dari aksara Jepang yang harus ditebak, misalnya "ka". Kemudian pengguna diharuskan menulis atau menggambarkan langsung pada kotak yang telah tersedia pada layar aplikasi menggunakan jari. Setelah selesai, maka pengguna harus menekan tombol "Recognize" yang berfungsi memerintahkan sistem untuk mengecek apakah tulisan cocok dengan aksara Jepang yang dimaksud.

D. Proses Pencocokkan

Sistem akan mencoba mendeteksi tulisan pengguna ke dalam bentuk aksara Jepang menggunakan *library* Tesseract. Jika sistem berhasil menemukan aksara yang mirip dengan tulisan tangan pengguna, maka sistem akan mencocokkan dengan aksara Jepang yang dijadikan sebagai soal.

IV. KESIMPULAN

Kemampuan perhitungan perangkat *mobile* berbasis Android semakin mendekati kemampuan komputer *workstation* sehingga dapat menjalankan Tesseract untuk mendeteksi citra tulisan tangan menjadi kedalam bentuk digital. Namun untuk keakuratan pencocokan pada Tesseract tergantung pada data training yang diberikan.

DAFTAR PUSTAKA

Smith R. 2007 . An Overview of the Tesseract OCR Engine . ICDAR '07 Proceedings of the Ninth International Conference on Document Analysis and Recognition II; 2007 Sept 23 - 26; Curitiba, Brasil. Washington DC (US): IEEE Computer Society. hlm 629 - 633.